



OTYŁOŚĆ

Alicja Cyganiewicz

alicja.cyganiewicz@infakt.pl

KRUG #6 / 2017

powered by Railsware.pl

17.10.2017 r.



Prawie trzy czwarte
amerykańskich mężczyzn i
ponad 60 procent kobiet cierpi
na otyłość lub nadwagę





Ryzyko chorób serca,
problemy z oddychaniem,
zwydrodnienia stawów...





Otyłość co roku zabija 300-400 tys. Amerykanów





OTYŁOŚĆ

?





OTYŁOŚĆ



245 kcal



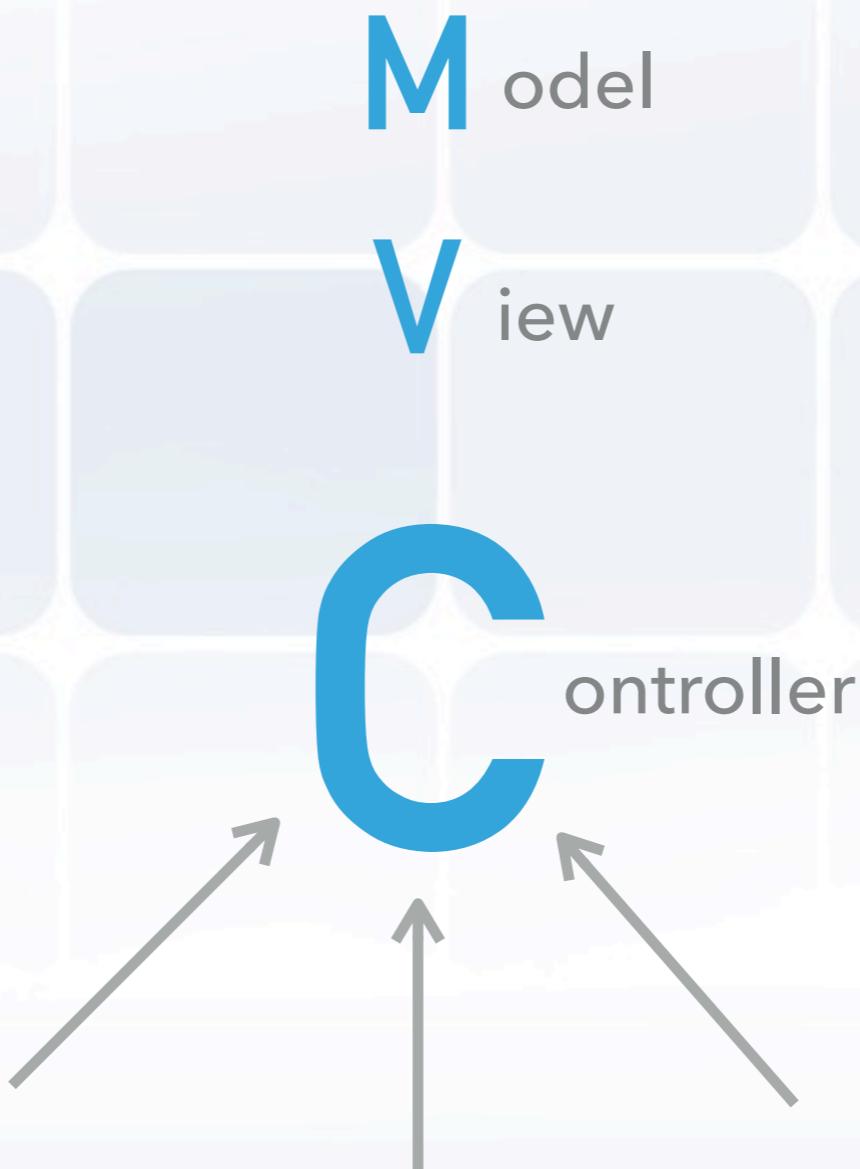


```
class BooksController << ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{Time.now.to_i}#{current_user.id}".to_shal[-6..-1]
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
    end
    render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
  end

  def update
    scope = current_user.books.not_kind(:draft).not_kind(:sketch)
      .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
    if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
      @search = scope.search({})
      @books = []
      @currency = 'PLN'
    else
      search = params[:search]
      @currency = search[:waluta_eq_any]
      search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'

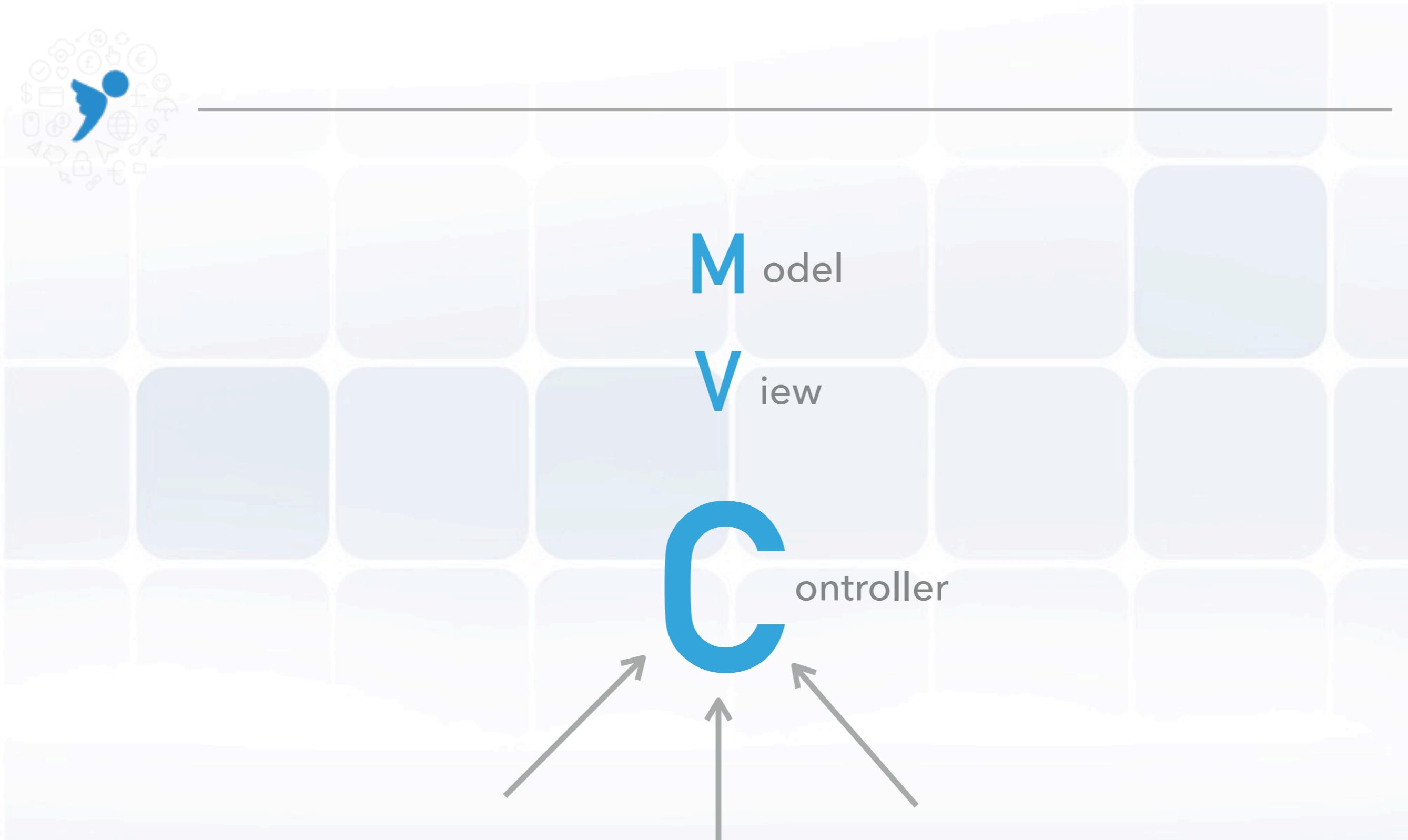
      if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
        search[:status_eq_any] += %w[wydrukowana wysÅ,ana]
        search[:status_eq_any].uniq!
      end
      @search = scope.search(params[:search])

      @books = @search.result(distinct: true)
    end
  end
end
```

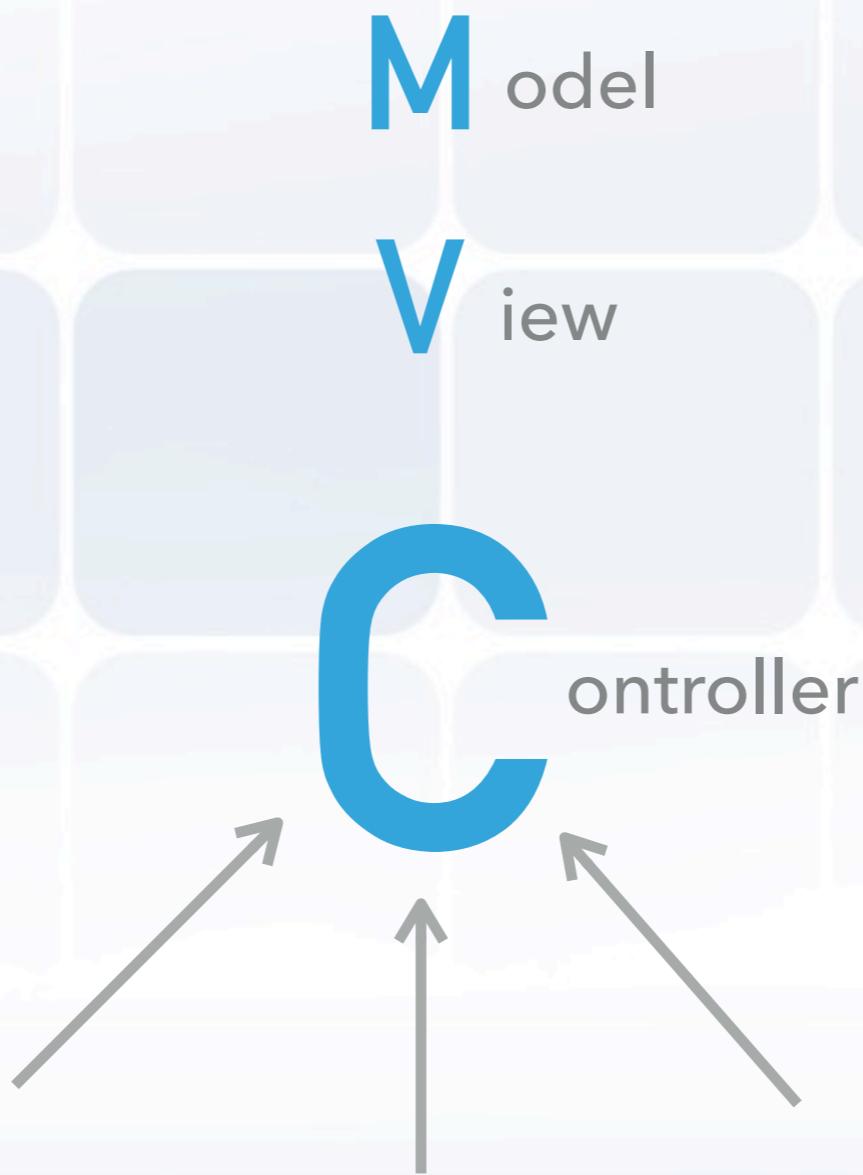
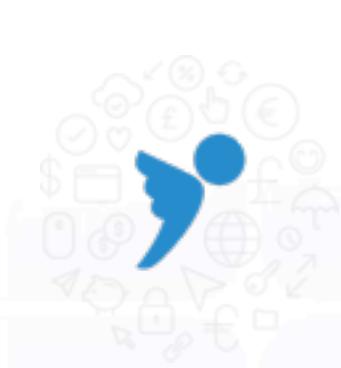


Logika Biznesowa

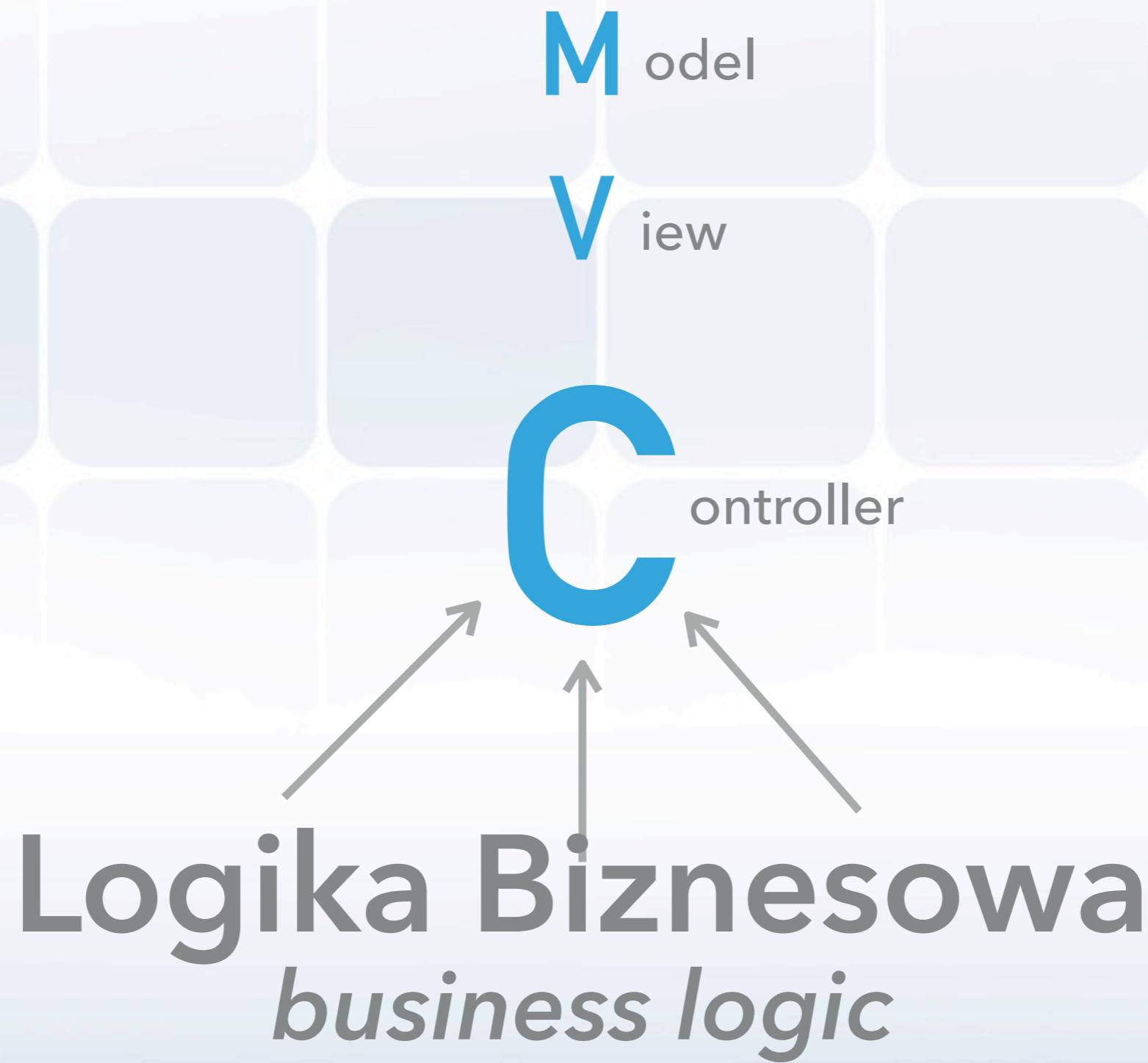
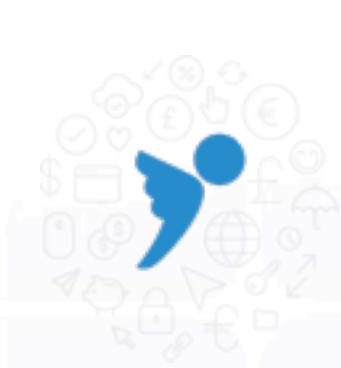
business logic

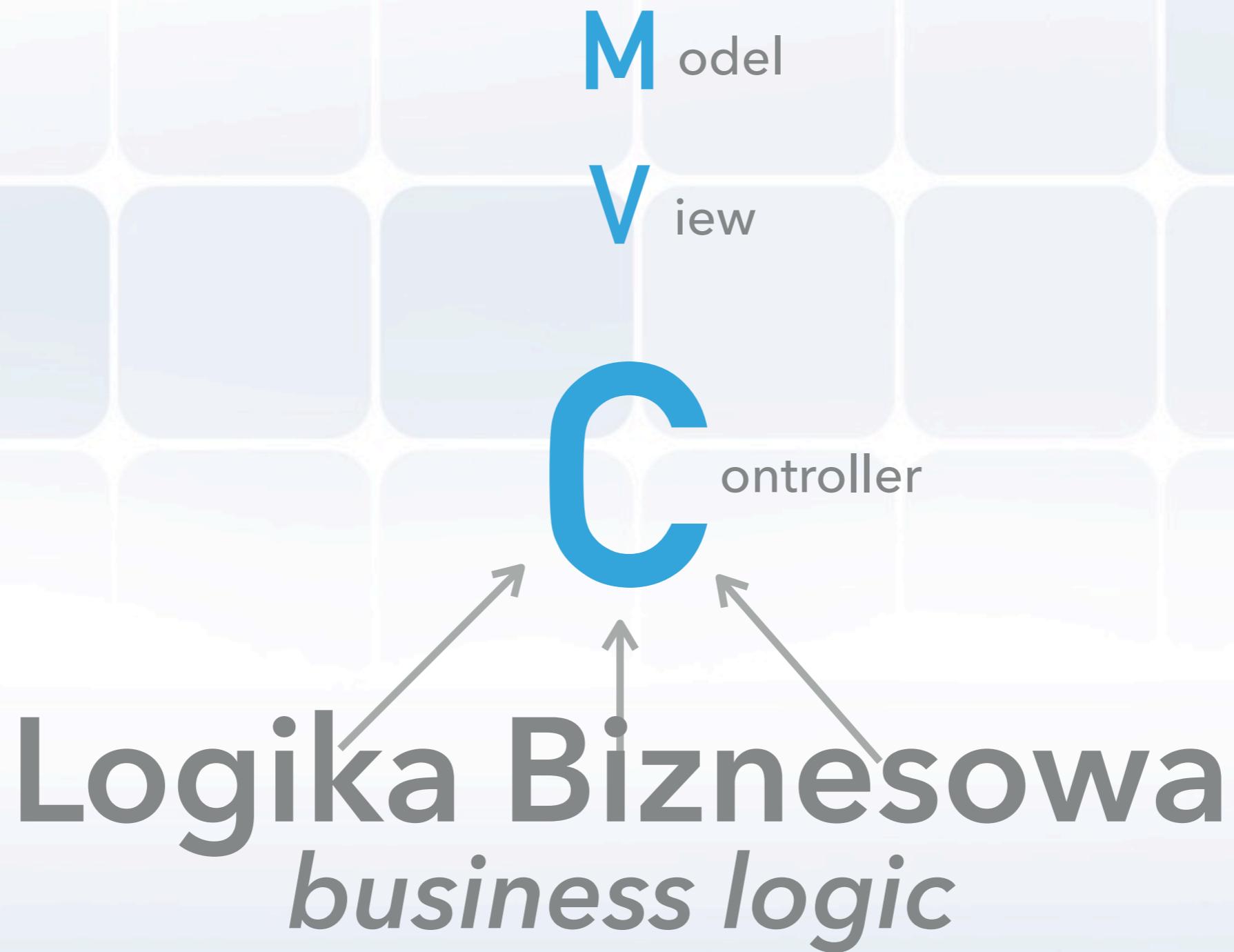


Logika Biznesowa
business logic



Logika Biznesowa
business logic







M odel

V iew

Logika Biznesowa *business logic*



Alicja Cyganiewicz

Ruby Dev at **inFakt**

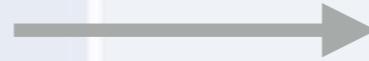
alicja.cyganiewicz@infakt.pl



Alicja Cyganiewicz

Ruby Dev at **inFakt**

alicja.cyganiewicz@infakt.pl



księgowość



Alicja Cyganiewicz

Ruby Dev at **inFakt**

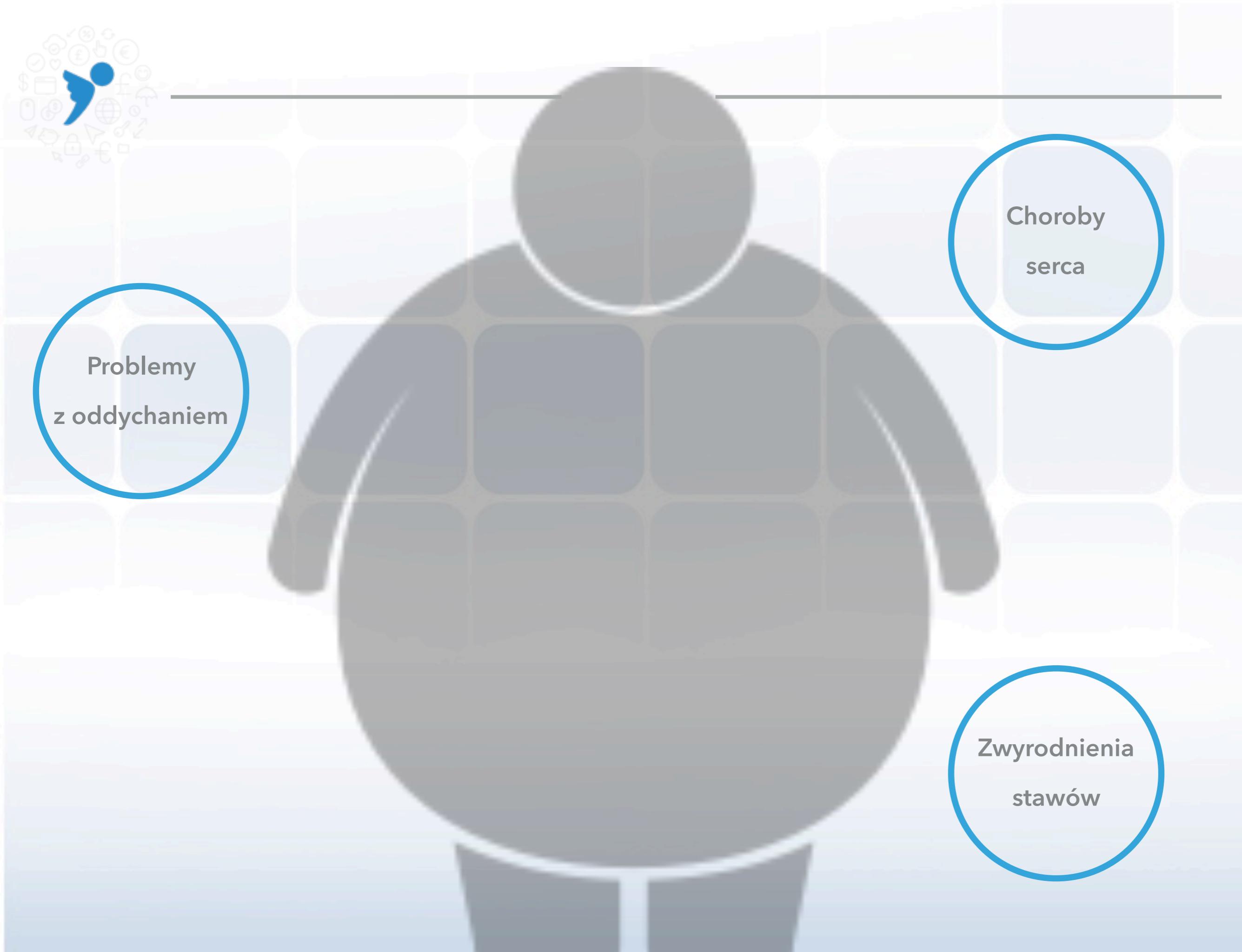
alicja.cyganiewicz@infakt.pl



księgowość



logika **biznesowa**



Problemy
z oddychaniem

Choroby
serca

Zwyrodnienia
stawów



Tests...

```
class BooksController << ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{Time.now.to_i}#{(current_user.id)}.to_sha1[-6..-1]"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
    end
    render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
  end

  def update
    scope = current_user.books.not_kind(:draft).not_kind(:sketch)
      .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
    if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
      @search = scope.search({})
      @books = []
      @currency = 'PLN'
    else
      search = params[:search]
      @currency = search[:waluta_eq_any]
      search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'

      if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
        search[:status_eq_any] += %w[wydrukowana wysÅ,ana]
        search[:status_eq_any].uniq!
      end
      @search = scope.search(params[:search])

      @books = @search.result(distinct: true)
    end
  end
end
```

DRY...?

Maintainable



Ponowne wykonanie logiki

np. API



DRY...?



Ponowne wykonanie logiki

np. API



~~CANCELLED~~
Don't

R epeat

Y ourself

DRY...?



controller.rb

```
class BooksController < < ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{$Time.now.to_i}#{$current_user.id}#{Time.now.to_i.to_s[-6..-1]}"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @search = scope.search({})
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'
    if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
      search[:status_eq_any] += %w[wydrukowana wysAana]
      search[:status_eq_any].uniq!
    end
    @search = scope.search(params[:search])
    @books = @search.result(distinct: true)
  end
end
```

controller_spec.rb

```
class BooksController < < ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{$Time.now.to_i}#{$current_user.id}#{Time.now.to_i.to_s[-6..-1]}"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'
    if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
      search[:status_eq_any] += %w[wydrukowana wysAana]
      search[:status_eq_any].uniq!
    end
    @search = scope.search(params[:search])
    @books = @search.result(distinct: true)
  end
end
```

Tests



controller.rb

```
class BooksController < ApplicationController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{$Time.now.to_i}#{$current_user.id}#{Time.now.to_f.to_s[-6..-1]}"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @search = scope.search({})
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'
  end
  if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
    search[:status_eq_any] += %w[wydrukowana wysAana]
    search[:status_eq_any].uniq!
  end
  @search = scope.search(params[:search])
  @books = @search.result(distinct: true)
end
end
```

controller_spec.rb

Tests

stub_all
test_nothing

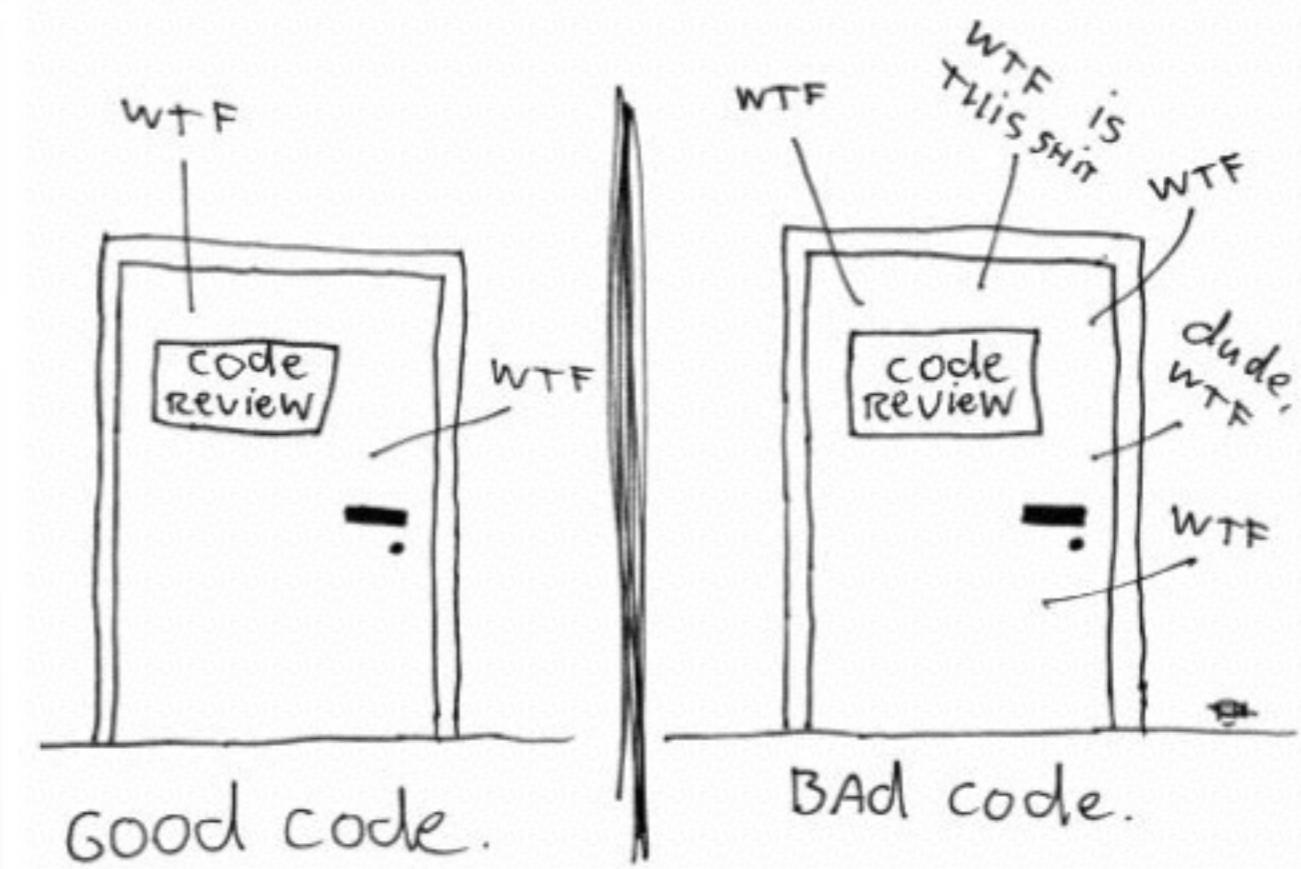


Kompletnie nieczytelny
kod

Nierozszerzalny
low maintainability

Maintainable

The ONLY VALID MEASUREMENT
OF CODE QUALITY: WTFs/MINUTE





There was a **time**...

Bad cake!



views
controllers
models

Good cake!



views
controllers
models



There was a **time**...

Skinny controllers, **FAT** models

Bad cake!



views
controllers
models

Good cake!

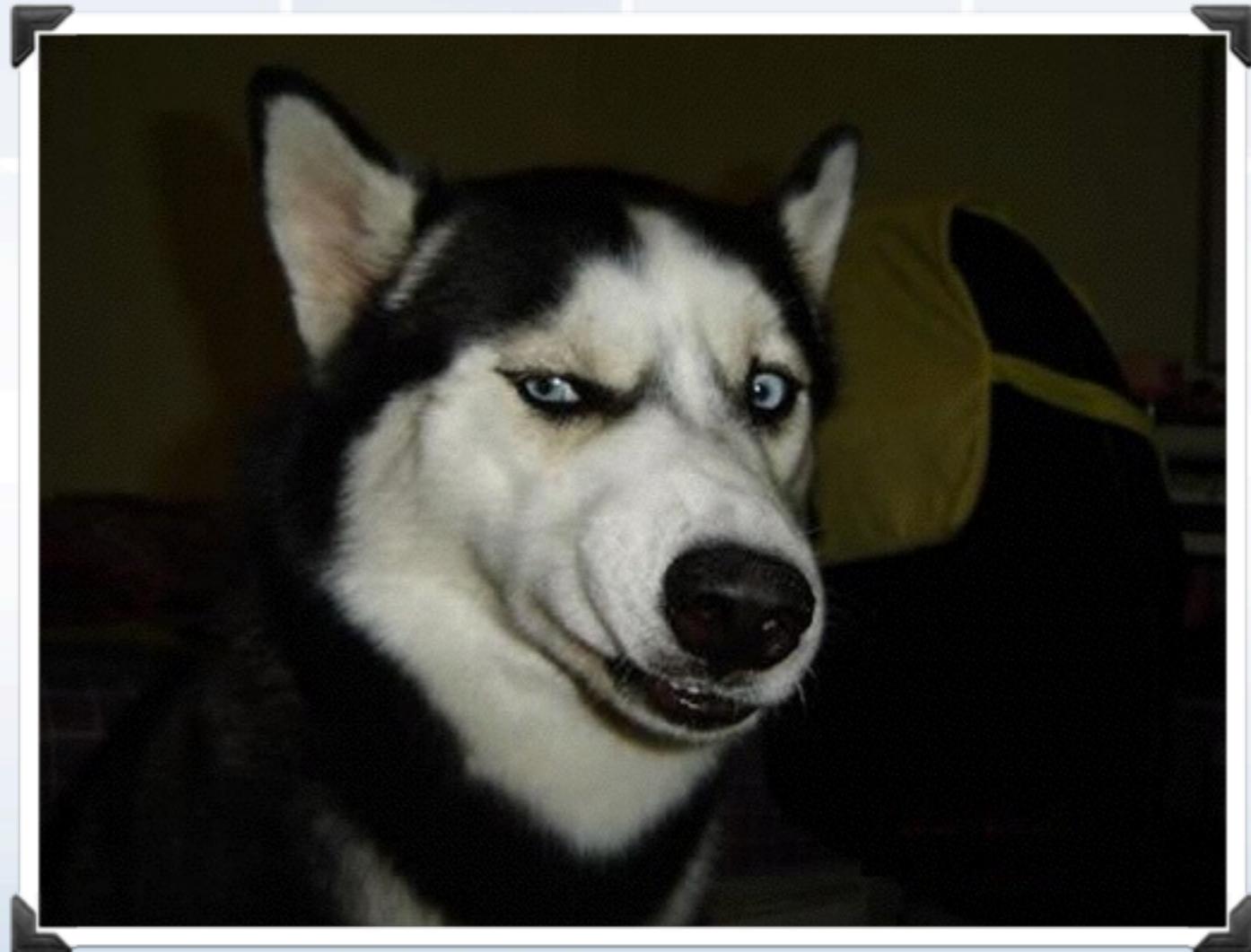


views
controllers
models



Skinny controllers, **FAT** models

Business **logic** in **models**?





So were we are?

FAT controllers

```
class BooksController < < ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#(Time.now.to_i)#{current_user.id}#{Time.now.to_i}_to_shal[-6..-1]"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @search = scope.search({})
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'

    if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
      search[:status_eq_any] += %w[wydrukowana wysAana]
      search[:status_eq_any].uniq!
    end
    @search = scope.search(params[:search])
  end
  @books = @search.result(distinct: true)
end
end
```



So were we **are**?

FAT controllers

```
class BooksController << ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#(Time.now.to_i)#{current_user.id}#{Time.now.to_i}_to_shal[-6..-1]"
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
    end
    render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
  end

  def update
    scope = current_user.books.not_kind(:draft).not_kind(:sketch)
    .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
    if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
      @search = scope.search({})
      @books = []
      @currency = 'PLN'
    else
      search = params[:search]
      @currency = search[:waluta_eq_any]
      search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'

      if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
        search[:status_eq_any] += %w[wydrukowana wysAana]
        search[:status_eq_any].uniq!
      end
      @search = scope.search(params[:search])
    end
    @books = @search.result(distinct: true)
  end
end
```

M odel

V iew

C ontroller



So were we **are**?

FAT controllers

```
class BooksController < < ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#{$(Time.now.to_i)}#{$(current_user.id)}".to_shal[-6..-1]
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @search = scope.search({})
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'
  end
  if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
    search[:status_eq_any] += %w[wydrukowana wysAana]
    search[:status_eq_any].uniq!
  end
  @search = scope.search(params[:search])
  @books = @search.result(distinct: true)
end
end
```

M odel
V iew
C ontroller


Homeless
business
logic



SERVICES OBJECTS

SLIMMING THERAPY

FAT controllers

```
class BooksController < ActionController::Base
  def edit
    js_data = {}
    content = nil
    is_suspected = false
    action = symbolized_document_action(params[:a])
    if available_accountings?
      objects = ::Loader.new(current_user, params[:o]).objects
      is_suspected = objects.any? do |object|
        ::Books::Events::Verifier.new(object, action, date: params[:date]).blocked?
      end
    end
    case
    when is_suspected && current_limits.cad_add_books?
      fid = "#Time.now.to_i#(current_user.id)".to_shal[-6..-1]
      text = t(action, scope: 'flash.accountings.verify', default: :default)
      js_data['CorrectionBox'] = fid
      content = render_to_string(partial: '/shared/box/partials/suspected',
                                 locals: { fid: fid, text: text }).strip
    when is_suspected
      flash[:error] = t('flash.accountings.verify.blocked')
      js_data['reload'] = true
    end
  end
  render json: { status: 'ok', suspected: is_suspected, content: content, jsData: js_data }
end

def update
  scope = current_user.books.not_kind(:draft).not_kind(:sketch)
  .order("books.data_wystawienia ASC, books.id ASC").includes(:kind)
  if params[:search].blank? || params[:search].all? { |k, v| v.blank? }
    @search = scope.search({})
    @books = []
    @currency = 'PLN'
  else
    search = params[:search]
    @currency = search[:waluta_eq_any]
    search[:waluta_eq_any] = '' if search[:waluta_eq_any] == 'ALL'

    if search[:status_eq_any].present? && search[:status_eq_any].include?('booked')
      search[:status_eq_any] += %w[wydrukowana wysAana]
      search[:status_eq_any].uniq!
    end
    @search = scope.search(params[:search])
  end
  @books = @search.result(distinct: true)
end
end
```



app

--- services

----- books

----- comment_creator.rb

```
1 module Books
2   class CommentCreator
3   end
4 end
5
```

Line 3, Column 6

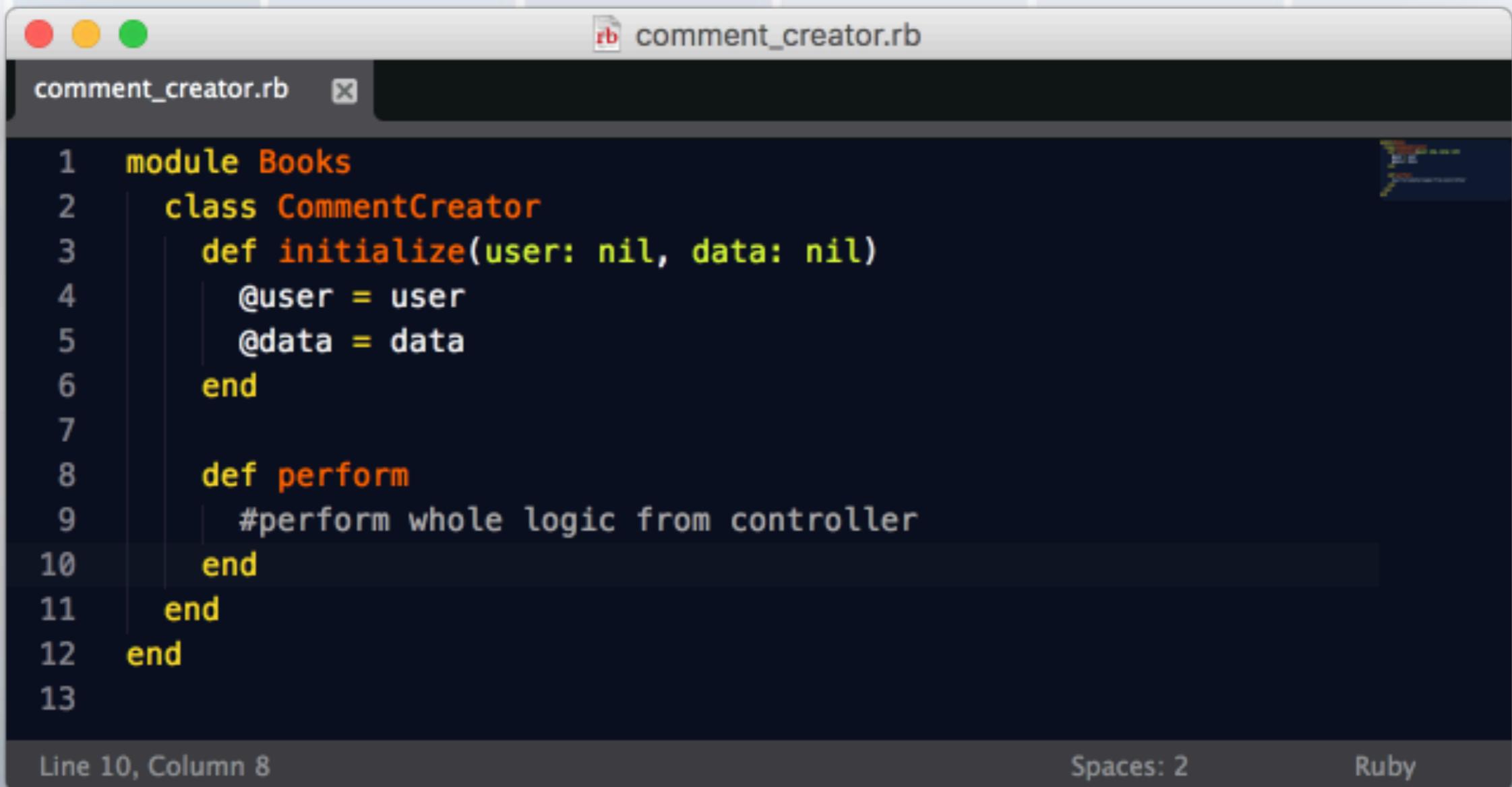
Spaces:

 app

--- services

---- books

----- comment_creator.rb



comment_creator.rb

```
1 module Books
2   class CommentCreator
3     def initialize(user: nil, data: nil)
4       @user = user
5       @data = data
6     end
7
8     def perform
9       #perform whole logic from controller
10    end
11  end
12 end
13
```

Line 10, Column 8 Spaces: 2 Ruby



```
comments_controller.rb
```

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5   def create
6     book = Book.find(params[:id])
7     if book && valid_form?
8       if current_user
9         nick = current_user.login
10      else
11        nick = "gosc" + Time.now.to_s
12      end
13      if book.comments.create(comment_data.merge(author: nick))
14        book.followers.each do |follower|
15          ::Books::CommentNotifier.send_email(follower)
16        end
17        Books::RateCalculatorWorker.perform_async(book.id)
18        redirect_to book_path(book)
19      else
20        #...

```

Line 29, Column 4 Spaces: 2 Ruby

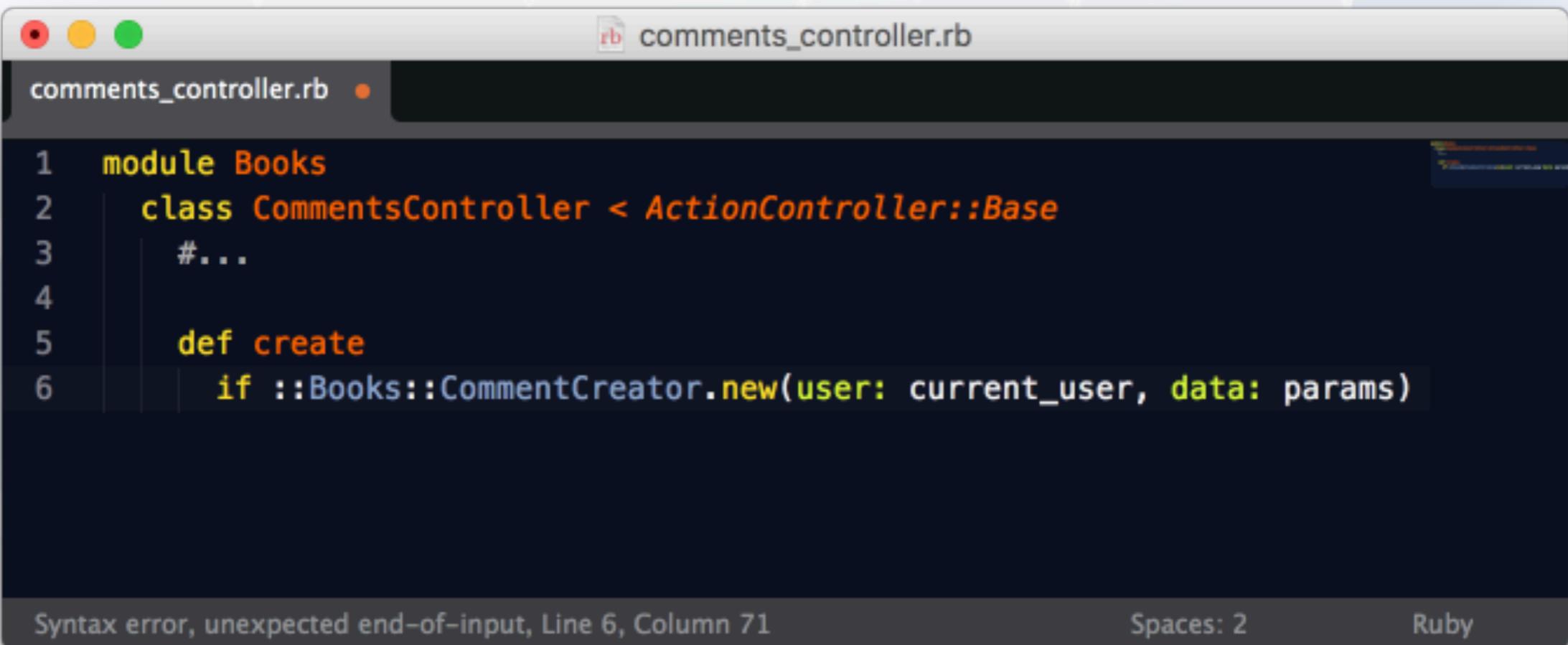


```
comments_controller.rb
```

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5   def create
6     if ::Books::CommentCreator
```

Syntax error, unexpected end-of-input, Line 6, Column 33

Spaces: 2 Ruby

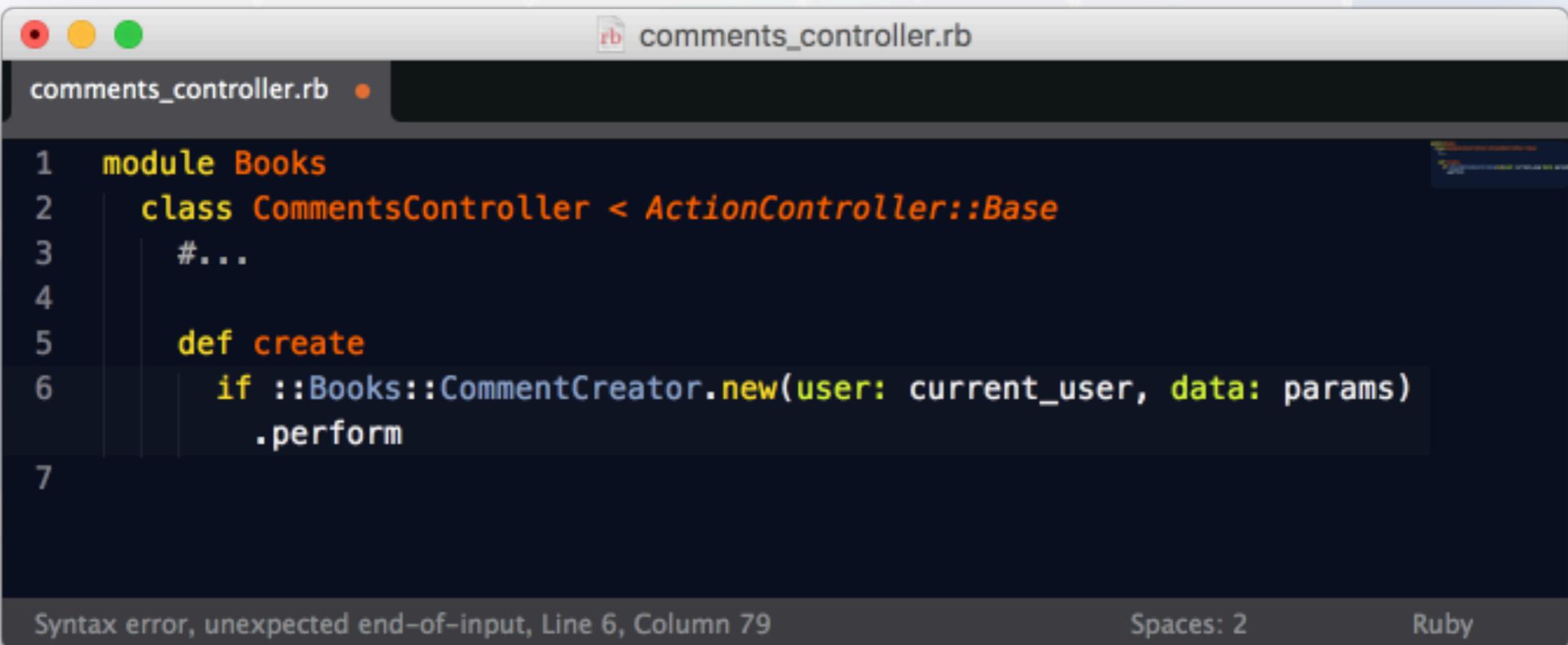


```
comments_controller.rb
```

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5   def create
6     if ::Books::CommentCreator.new(user: current_user, data: params)
```

Syntax error, unexpected end-of-input, Line 6, Column 71

Spaces: 2 Ruby



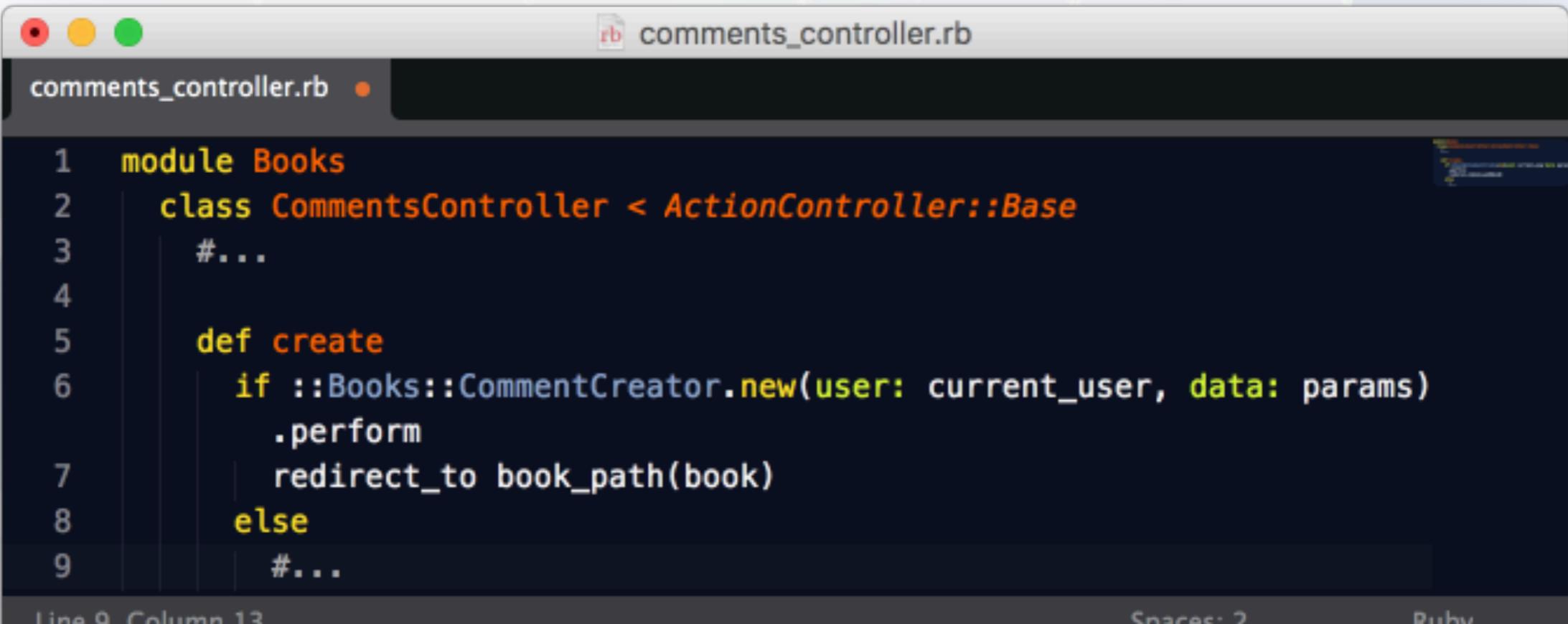
```
comments_controller.rb
```

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5     def create
6       if ::Books::CommentCreator.new(user: current_user, data: params)
7         .perform
8       end
9     end
10  end
```

Syntax error, unexpected end-of-input, Line 6, Column 79

Spaces: 2

Ruby



```
comments_controller.rb •
```

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5     def create
6       if ::Books::CommentCreator.new(user: current_user, data: params)
7         .perform
8         redirect_to book_path(book)
9       else
10       #...
11     end
12   end
13 end
```

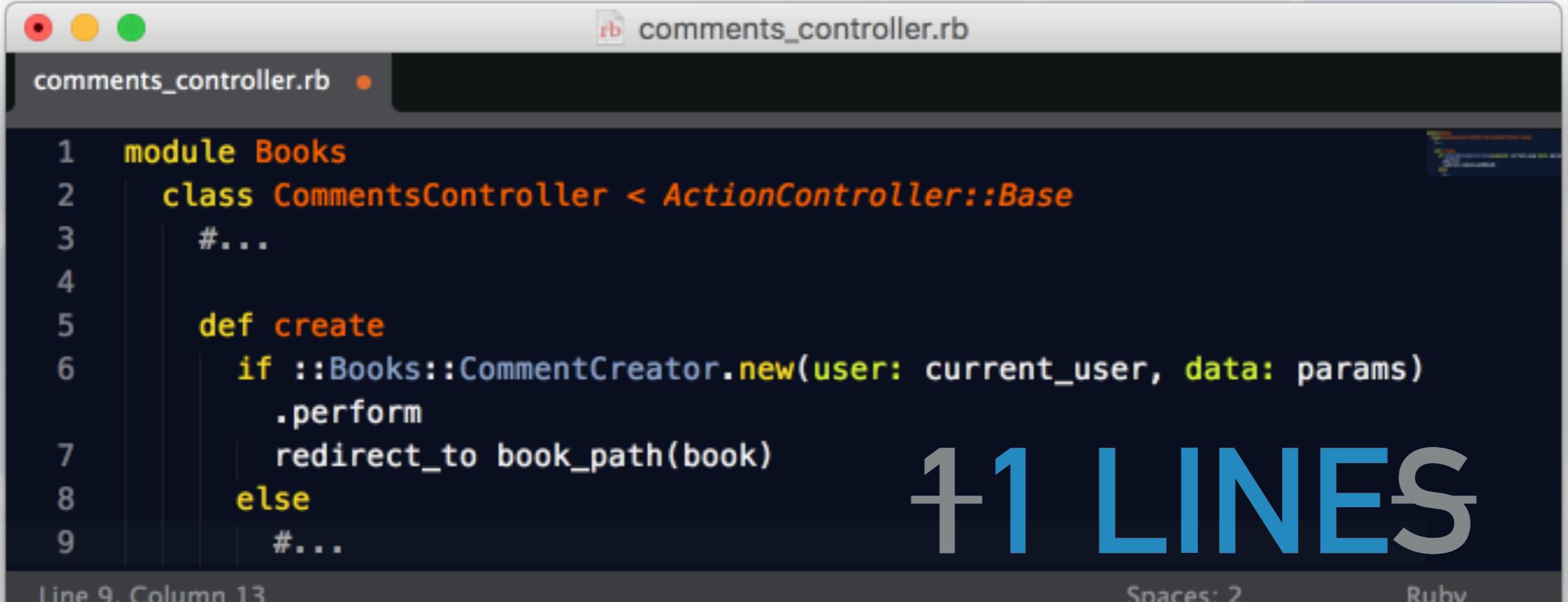
Line 9, Column 13 Spaces: 2 Ruby





A screenshot of a Ruby code editor showing the file `comments_controller.rb`. The code defines a `CommentsController` class within the `Books` module, handling a `create` action. It finds a book by ID, checks if it's valid, and creates a comment for the book. If the user is logged in, the comment is attributed to their login; otherwise, it uses a timestamp-based nick. It then sends notifications to followers and updates a rate calculator worker. The code is annotated with line numbers from 1 to 20.

```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5   def create
6     book = Book.find(params[:id])
7     if book && valid_form?
8       if current_user
9         nick = current_user.login
10      else
11        nick = "gosc" + Time.now.to_s
12      end
13      if book.comments.create(comment_data.merge(author: nick))
14        book.followers.each do |follower|
15          ::Books::CommentNotifier.send_email(follower)
16        end
17        Books::RateCalculatorWorker.perform_asynch(book.id)
18        redirect_to book_path(book)
19      else
20        #...
21      end
22    end
23  end
24
25  private
26    def comment_data
27      params.require(:comment).permit(:body)
28    end
29  end
```



comments_controller.rb

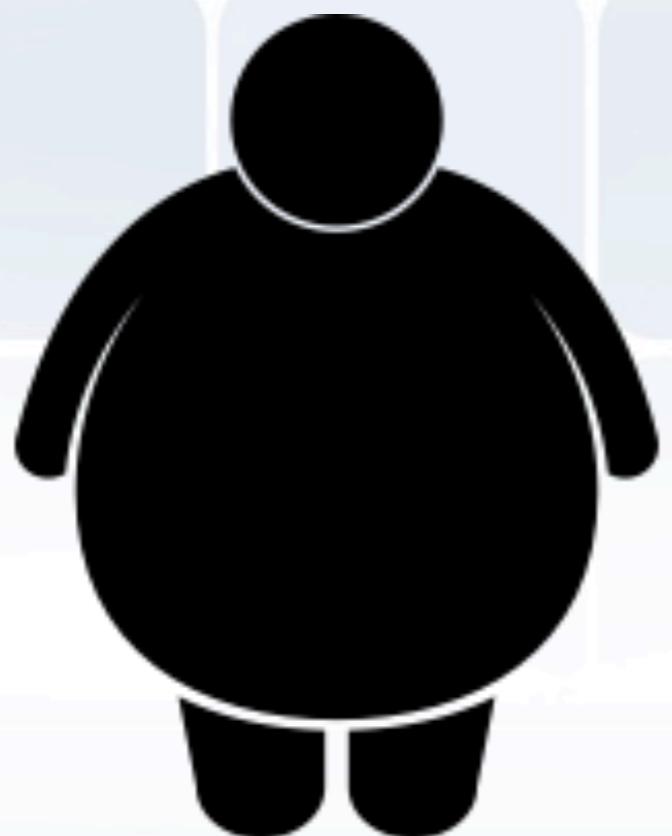
```
1 module Books
2   class CommentsController < ActionController::Base
3     #...
4
5     def create
6       if ::Books::CommentCreator.new(user: current_user, data: params)
7         .perform
8         redirect_to book_path(book)
9       else
10         #...
11     end
12   end
13 end
```

Line 9, Column 13

11 LINES

Spaces: 2

Ruby



Services Objects
→
slimming therapy



```
comments_controller.rb • comments_controller.rb  
1 module Books  
2   class CommentsController < ActionController::Base  
3     #...  
4  
5     def create  
6       if ::Books::CommentCreator.new(user: current_user, data: params)  
7         .perform  
8         redirect_to book_path(book)  
9       else  
#...  
Line 9, Column 13 Spaces: 2 Ruby
```

DRY...?

Service re-used!

```
comments_controller.rb • comments_controller.rb  
1 module Api  
2   module Books  
3     class CommentsController < ::Api::BaseController  
4       #...  
5  
6       def create  
7         if ::Books::CommentCreator.new(user: current_user, data:  
8           params).perform  
9           render json: { comment: comment }, status: :created  
#...  
Line 9, Column 13 Spaces: 2 Ruby
```

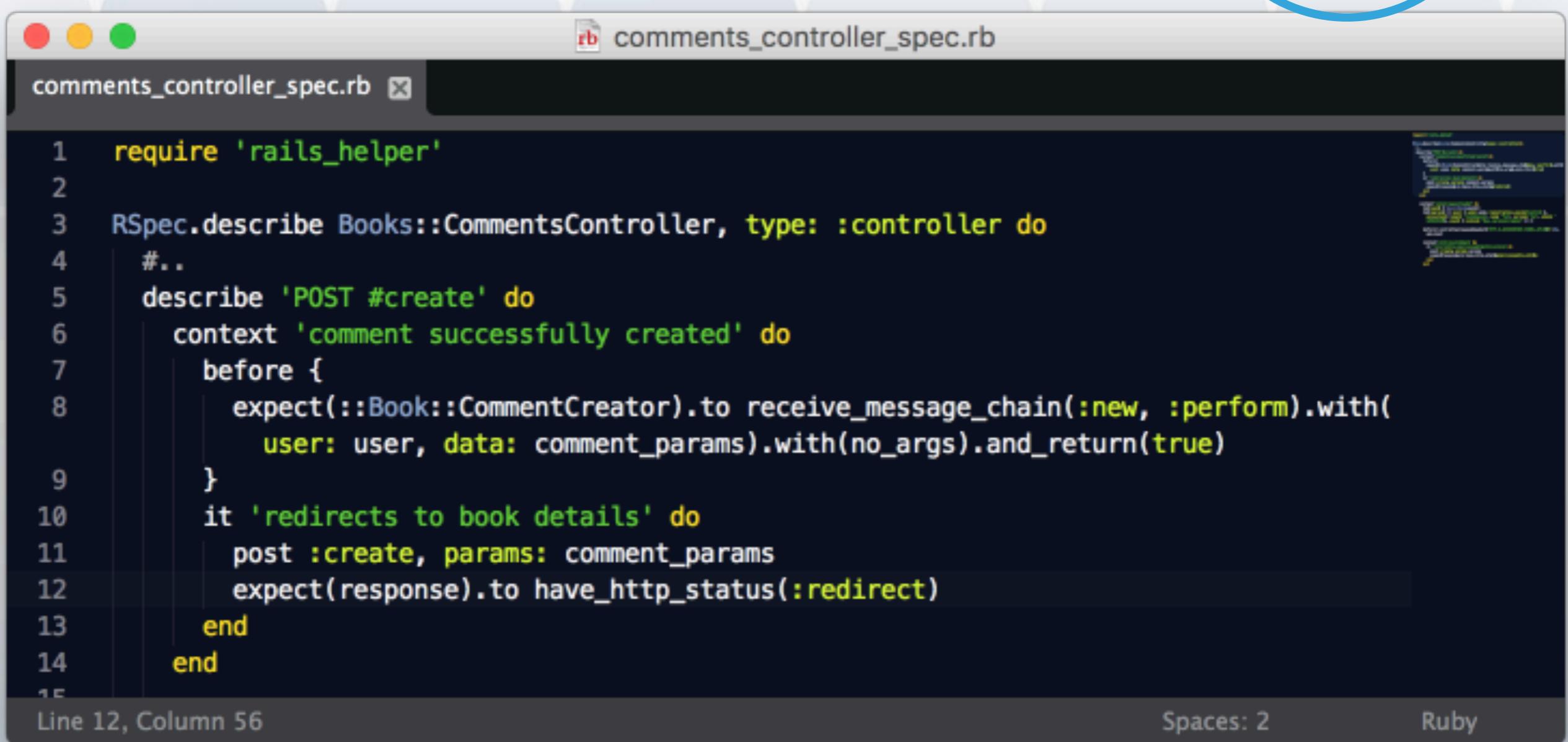


**prosty test controllera testujący to,
czego od niego oczekujemy**

Tests

prosty test controllera testujący to, czego od niego oczekujemy

Tests



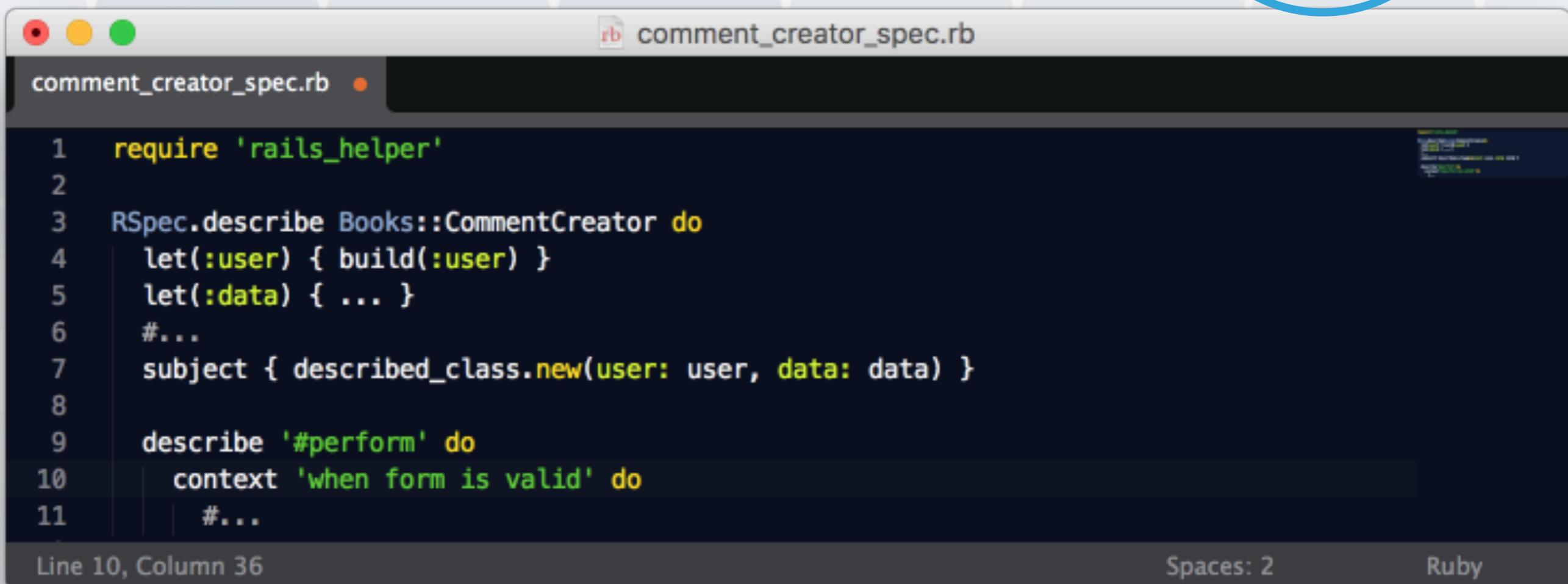
```
1 require 'rails_helper'
2
3 RSpec.describe Books::CommentsController, type: :controller do
4   #..
5   describe 'POST #create' do
6     context 'comment successfully created' do
7       before {
8         expect(::Book::CommentCreator).to receive_message_chain(:new, :perform).with(
9           user: user, data: comment_params).with(no_args).and_return(true)
10      }
11      it 'redirects to book details' do
12        post :create, params: comment_params
13        expect(response).to have_http_status(:redirect)
14      end
15    end

```

Line 12, Column 56 Spaces: 2 Ruby

...a całą skomplikowaną logikę przetestujemy sobie
spokojnie w **teście serwisu**

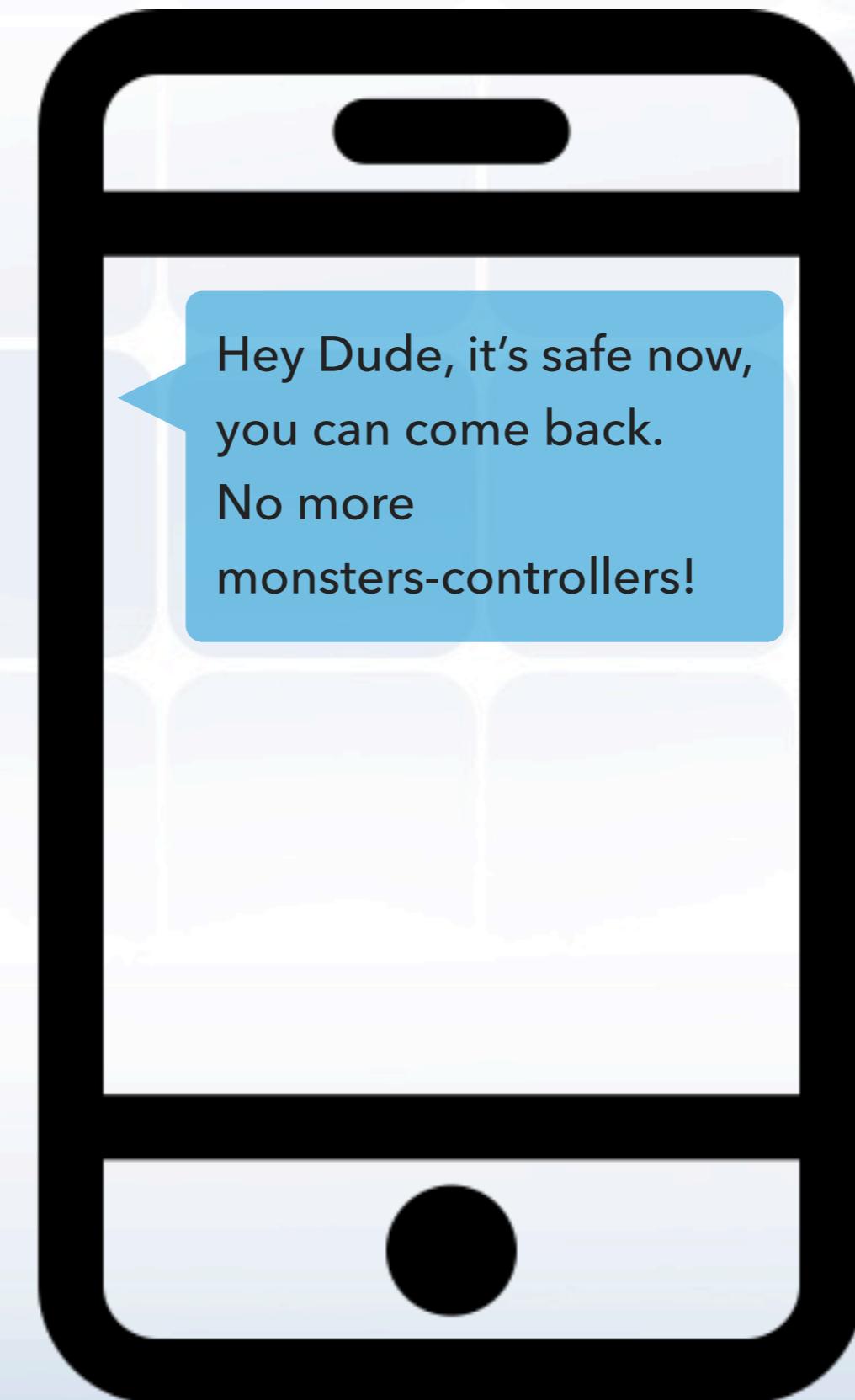
Tests

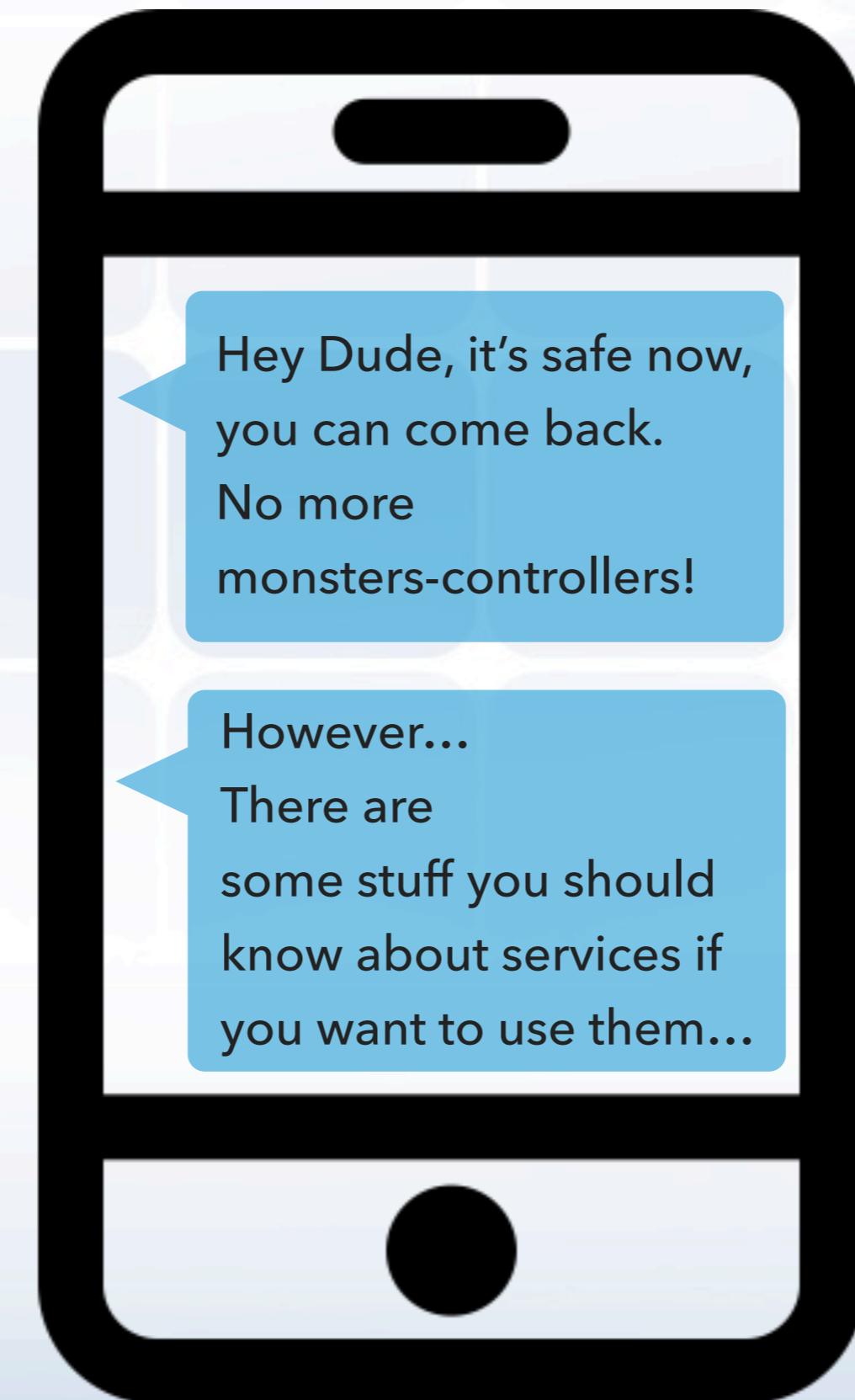


```
comment_creator_spec.rb
```

```
1 require 'rails_helper'  
2  
3 RSpec.describe Books::CommentCreator do  
4   let(:user) { build(:user) }  
5   let(:data) { ... }  
6   #...  
7   subject { described_class.new(user: user, data: data) }  
8  
9   describe '#perform' do  
10     context 'when form is valid' do  
11       #...
```

Line 10, Column 36 Spaces: 2 Ruby





Plain

Old

Ruby

Object

SERVICES OBJECTS



RAILS

S_ingle R_esponsibility P_rinciple



Single Responsibility Principle
Just because you *can* doesn't mean you *should*.

Self - explanatory

naming

MoneyConverter,

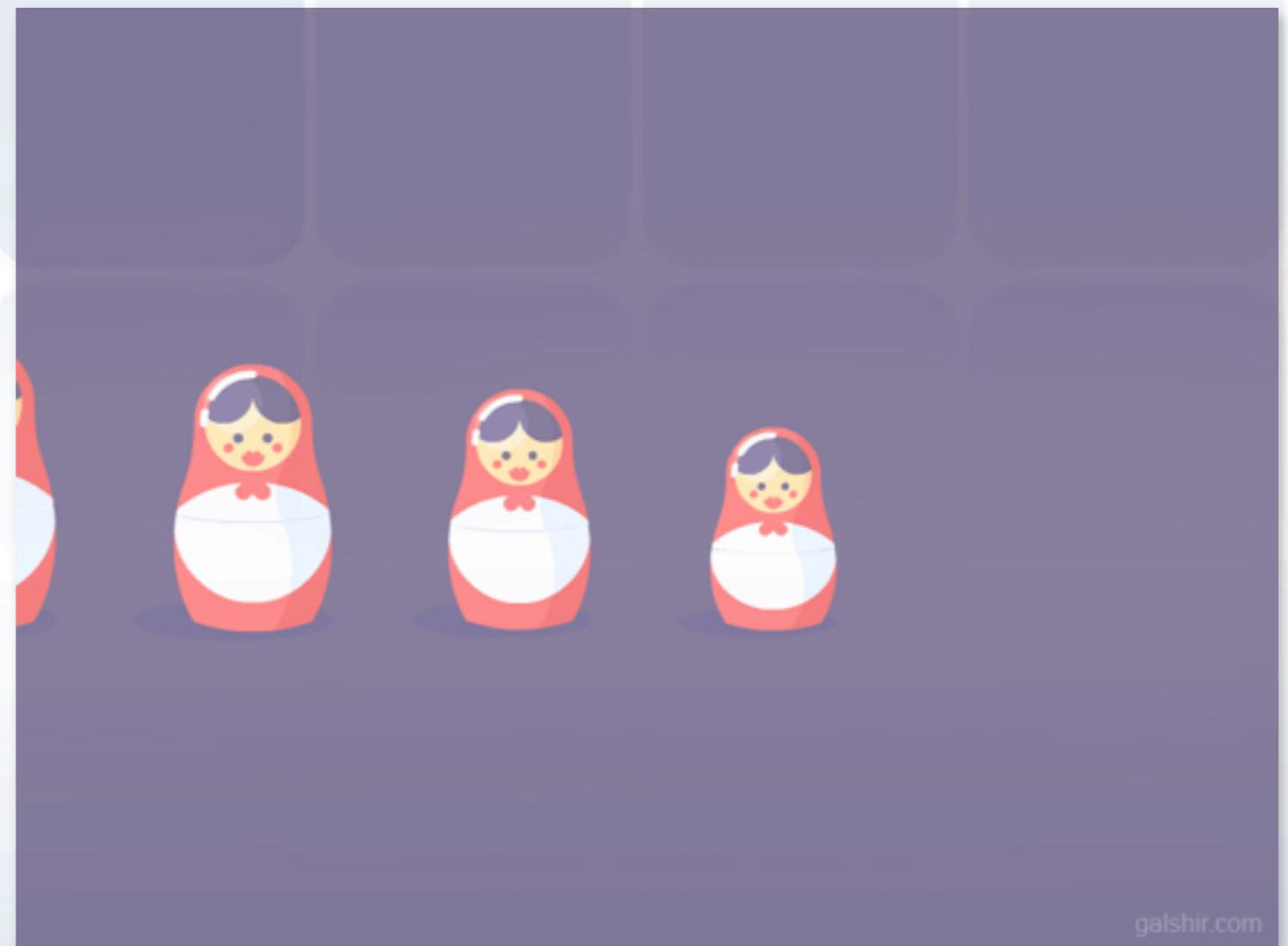
AgreementSigner,

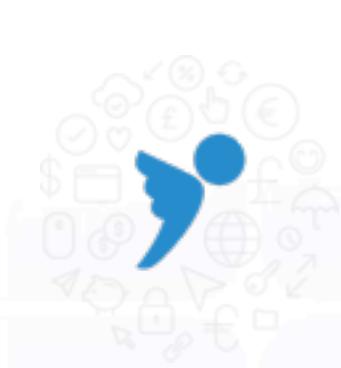
EventFetcher...

One public
method

SRP → .PERFORM

Services in a service







books_controller.rb

```
1 class BooksController < ActionController::Base
2   #...
3   def update
4     @book = Book.find(params[:id])
5     if ::BookUpdater.new(@book, params).perform
6       redirect_to book_path(@book)
7     else
8       render 'edit'
9     end
10    end
11  end
12
```

rb books_controller.rb

Line 1, Column 1

Spaces: 2

Ruby



books_controller.rb

```
1 class BooksController < ActionController::Base
2   #...
3   def update
4     @book = Book.find(params[:id])
5     @form = ::BookForm.new(params)
6     if ::BookUpdater.new(@book, @form).perform
7       redirect_to book_path(@book)
8     else
9       render 'edit'
10    end
11  end
12 end
13
```

Line 5, Column 35

Spaces: 2

Ruby



books_controller.rb

```
1 class BooksController < ActionController::Base
2   #...
3   def update
4     @book = Book.find(params[:id])
5     @form = ::BookForm.new(params)
6     @presenter = ::BookPresenter.new(@book)
7     if ::BookUpdater.new(@book, @form).perform
8       redirect_to book_path(@book)
9     else
10      render 'edit'
11    end
12  end
13 end
14
```

Line 6, Column 44

Spaces: 2

Ruby



books_controller.rb

```
1 class BooksController < ActionController::Base
2   #...
3   def update
4     @book = Book.find(params[:id])
5     @form = ::BookForm.new(params)
6     @presenter = ::BookPresenter.new(@book)
7     if ::BookUpdater.new(@book, @form).perform
8       redirect_to book_path(@book)
9     else
10      if form.valid?
11        render 'update_error'
12      else
13        render 'edit'
14      end
15    end
16  end
17 end
18
```

Line 14, Column 10

Spaces: 2

Ruby



books_controller.rb

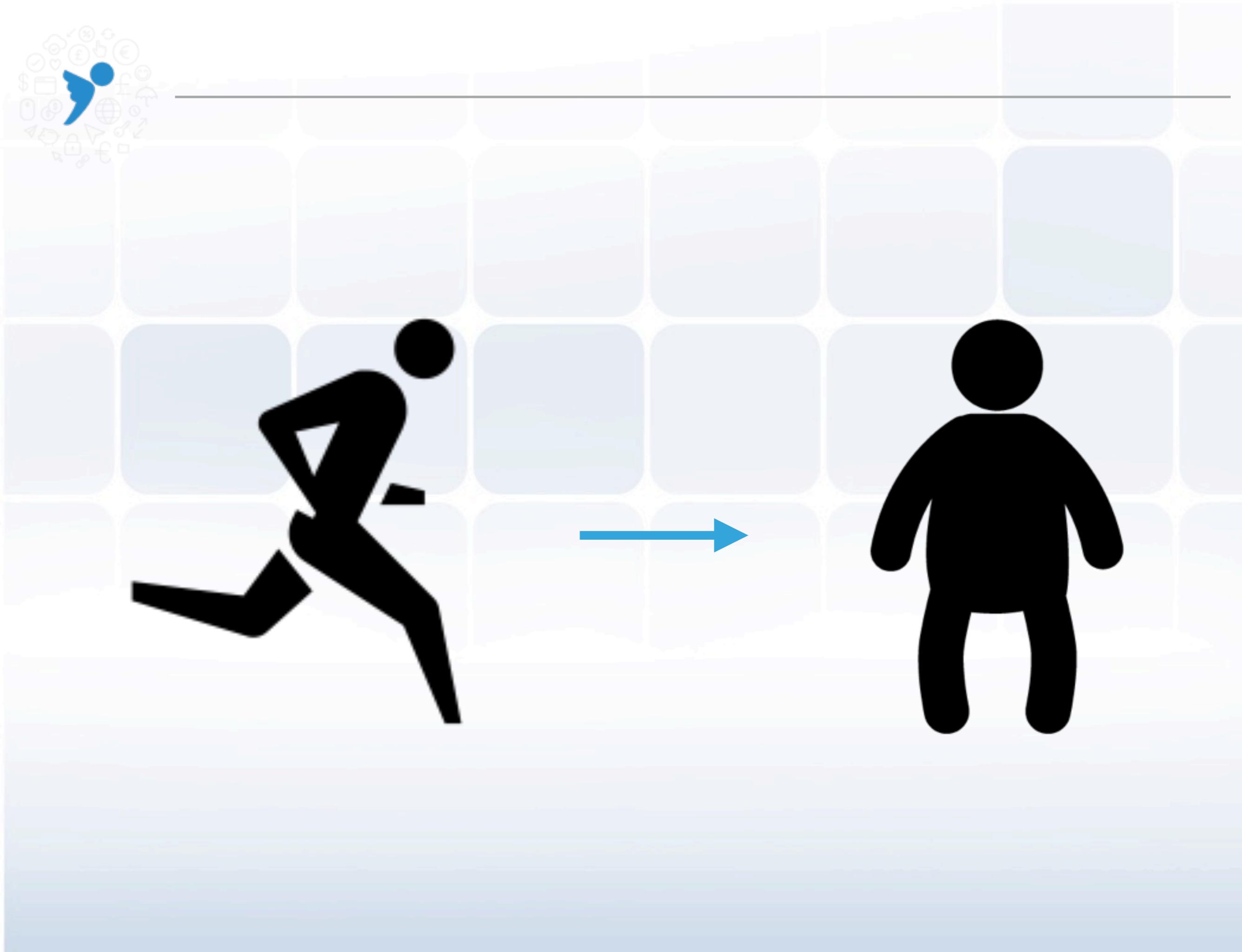
```
1 class BooksController < ActionController::Base
2   #...
3   def update
4     @book = Book.find(params[:id])
5     @form = build_form(params)
6     @presenter = ::BookPresenter.new(@book)
7     if ::BookUpdater.new(@book, @form).perform
8       redirect_to book_path(@book)
9     else
10      if form.valid?
11        render 'update_error'
12      else
13        render 'edit'
14      end
15    end
16  end
17
18  private
19
20  def build_form(form_params)
21    current_user.admin? ? ::Books::AdminForm.new(form_params) : ::Books::
22      UserForm.new(form_params)
23  end
```

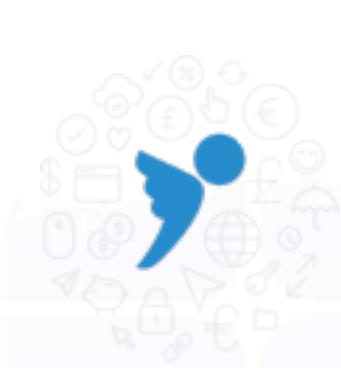


A screenshot of a code editor showing a Ruby file named `books_controller.rb`. The code defines a `BooksController` class with methods for `edit` and `update`, and a private method `build_form`. The code uses Active Record (`Book`), presenter (`BookPresenter`), and updater (`BookUpdater`) objects.

```
books_controller.rb

1  class BooksController < ActionController::Base
2    ...
3
4    def edit
5      @book = Book.find(params[:id])
6      @form = build_form(default_params)
7      @presenter = ::BookPresenter.new(@book)
8      render 'edit'
9    end
10
11   def update
12     @book = Book.find(params[:id])
13     @form = build_form(params)
14     @presenter = ::BookPresenter.new(@book)
15     if ::BookUpdater.new(@book, @form).perform
16       redirect_to book_path(@book)
17     else
18       if form.valid?
19         render 'update_error'
20       else
21         render 'edit'
22       end
23     end
24   end
25
26   private
27
28   def build_form(form_params)
29     current_user.admin? ? ::Books::AdminForm.new(form_params) : ::Books::UserForm.new(form_params)
30   end
31 end
32
```





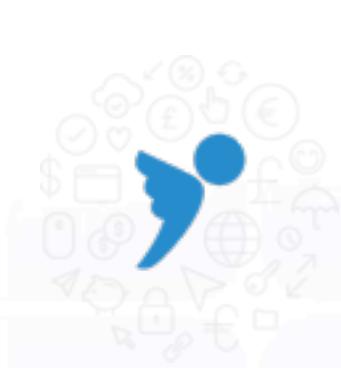
PRESENTER

FORM
OBJECT

UPDATER

OBJECT
(BOOK)

PARTIAL



CONTEXT

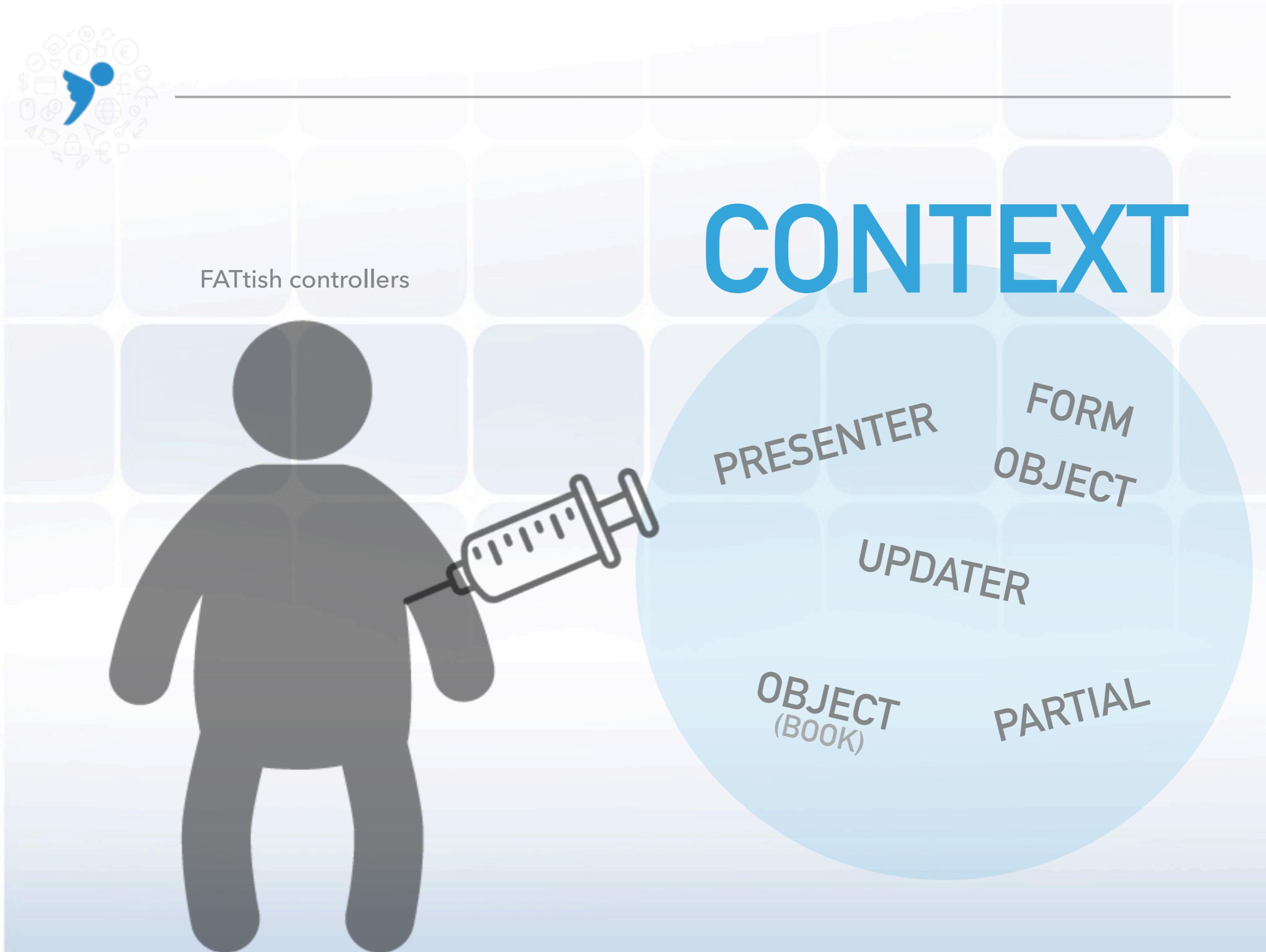
PRESENTER

FORM
OBJECT

UPDATER

OBJECT
(BOOK)

PARTIAL



CONTEXT

FATtish controllers

PRESENTER

FORM
OBJECT

UPDATER

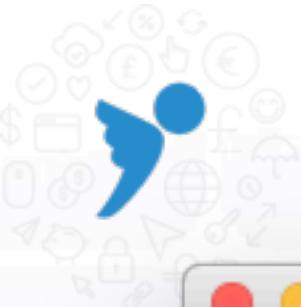
OBJECT
(BOOK)

PARTIAL



CONTEXTS
DIET

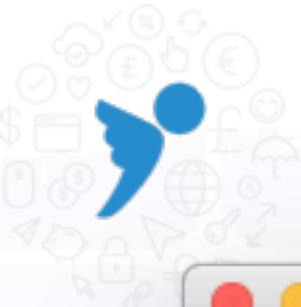




```
context.rb context.rb  
context.rb  
1 class BookContext  
2  
3   def initialize(user: nil, params: {})  
4     @user = user  
5     @params = params  
6   end  
7 end  
8
```

app
--- services
----- book_updater.rb
----- book_context.rb

Line 7, Column 4 Spaces: 2 Ruby



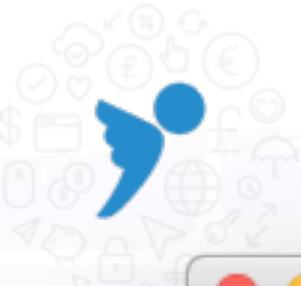
```
context.rb context.rb

1 class BookContext
2
3   def initialize(user: nil, params: {})
4     @user = user
5     @params = params
6   end
7
8   def book
9     @book ||= Book.find(params[:id])
10  end
11
12  private
13  attr_reader :user, :params
14 end
15
```

Line 15, Column 1

Spaces: 2

Ruby



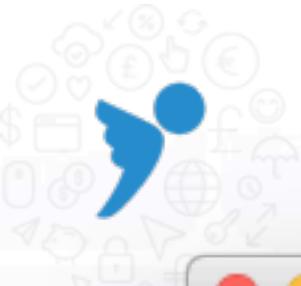
```
context.rb
```

```
1 class BookContext
2
3   def initialize(user: nil, params: {})
4     @user = user
5     @params = params
6   end
7
8   def book
9     @book ||= Book.find(params[:id])
10  end
11
12  def form
13    @form ||= form_klass.new(form_params)
14  end
15
16  private
17  attr_reader :params, :user
18
19  def form_klass
20    user.admin? ? ::Books::AdminForm : ::Books::UserForm
21  end
22
23  def form_params
24    default_params.merge(params[:form_data])
25  end
26 end
27
```

Line 1, Column 1

Spaces: 2

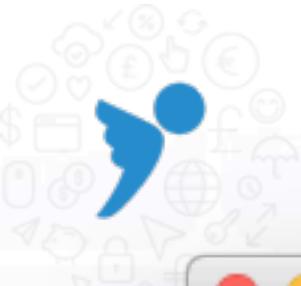
Ruby



```
context.rb context.rb
1 class BookContext
2
3   def initialize(user: nil, params: {})
4     @user = user
5     @params = params
6   end
7
8   def book
9     @book ||= Book.find(params[:id])
10  end
11
12  def form
13    @form ||= form_klass.new(form_params)
14  end
15
16  def updater
17    @updater ||= ::BookUpdater.new(book, form)
18  end
19
20  private
21  attr_reader :params, :user
22
23  def form_klass
24    user.admin? ? ::Books::AdminForm : ::Books::UserForm
25  end
26
27  def form_params
28    default params.merges(params[:form_data])

```

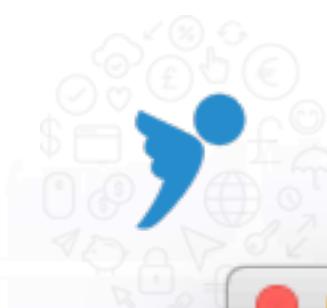
Line 18, Column 6 Spaces: 2 Ruby



```
context.rb context.rb
1 class BookContext
2
3   def initialize(user: nil, params: {})
4     @user = user
5     @params = params
6   end
7
8   def book
9     @book ||= Book.find(params[:id])
10  end
11
12  def form
13    @form ||= form_klass.new(form_params)
14  end
15
16  def updater
17    @updater ||= ::BookUpdater.new(book, form)
18  end
19
20  def presenter
21    @presenter ||= ::BookPresenter.new(book)
22  end
23
24  private
25  attr_reader :params, :user
26
27  def form_klass
28    user.admin? ? ::Books::AdminForm : ::Books::UserForm

```

Line 21, Column 45 Spaces: 2 Ruby



A large blue curly brace is positioned on the right side of the slide, spanning vertically from the code area to the text "USER CONTEXT, ECOSYSTEM".

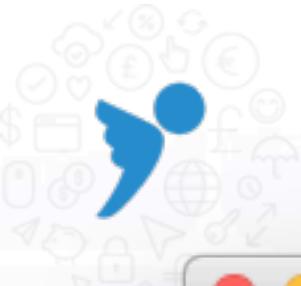
```
context.rb
```

```
1 class BookContext
2
3   def initialize(user: nil, params: {})
4     @user = user
5     @params = params
6   end
7
8   def book
9     @book ||= Book.find(params[:id])
10  end
11
12  def form
13    @form ||= form_klass.new(form_params)
14  end
15
16  def updater
17    @updater ||= ::BookUpdater.new(book, form)
18  end
19
20  def presenter
21    @presenter ||= ::BookPresenter.new(book)
22  end
23
24  private
25  attr_reader :params, :user
26
27  def form_klass
28    user.admin? ? ::Books::AdminForm : ::Books::UserForm

```

Line 21, Column 45 Spaces: 2 Ruby

USER CONTEXT, ECOSYSTEM



```
books_controller.rb
```

```
books_controller.rb
```

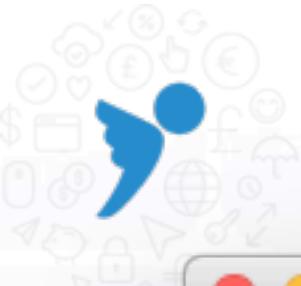
```
1 class BooksController < ActionController::Base
2   #...
3
4   def edit
5     @book = Book.find(params[:id])
6     @form = build_form(default_params)
7     @presenter = ::BookPresenter.new(@book)
8     render 'edit'
9   end
10
11  def update
12    @book = Book.find(params[:id])
13    @form = build_form(params)
14    @presenter = ::BookPresenter.new(@book)
15    if ::BookUpdater.new(@book, @form).perform
16      redirect_to book_path(@book)
17    else
18      if form.valid?
19        render 'update_error'
20      else
21        render 'edit'
22      end
23    end
24  end
25
26  private
27
28  def build_form(form_params)
29    current_user.admin? ? ::Books::AdminForm.new(form_params) : ::Books::UserForm.new(form_params)
30  end
31
32
```

Line 1, Column 1

Spaces: 2

Ruby

31 LINES



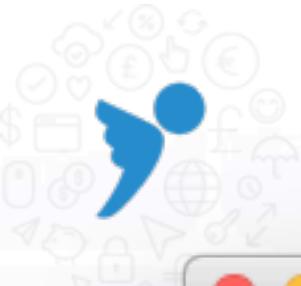
```
books_controller.rb
```

```
books_controller.rb
```

```
1 class BooksController < ActionController::Base
2   before_action :load_context
3   ...
4
5   def edit; end
6
7   def update
8     @context.updater.perform ? redirect_to(book_path(@context.book)) : render(@context.partial_path)
9   end
10
11  private
12
13  def load_context
14    @context = ::BookContext.new(user: current_user, params: params)
15  end
16 end
17
```

316 LINES

Line 17, Column 1 Spaces: 2 Ruby



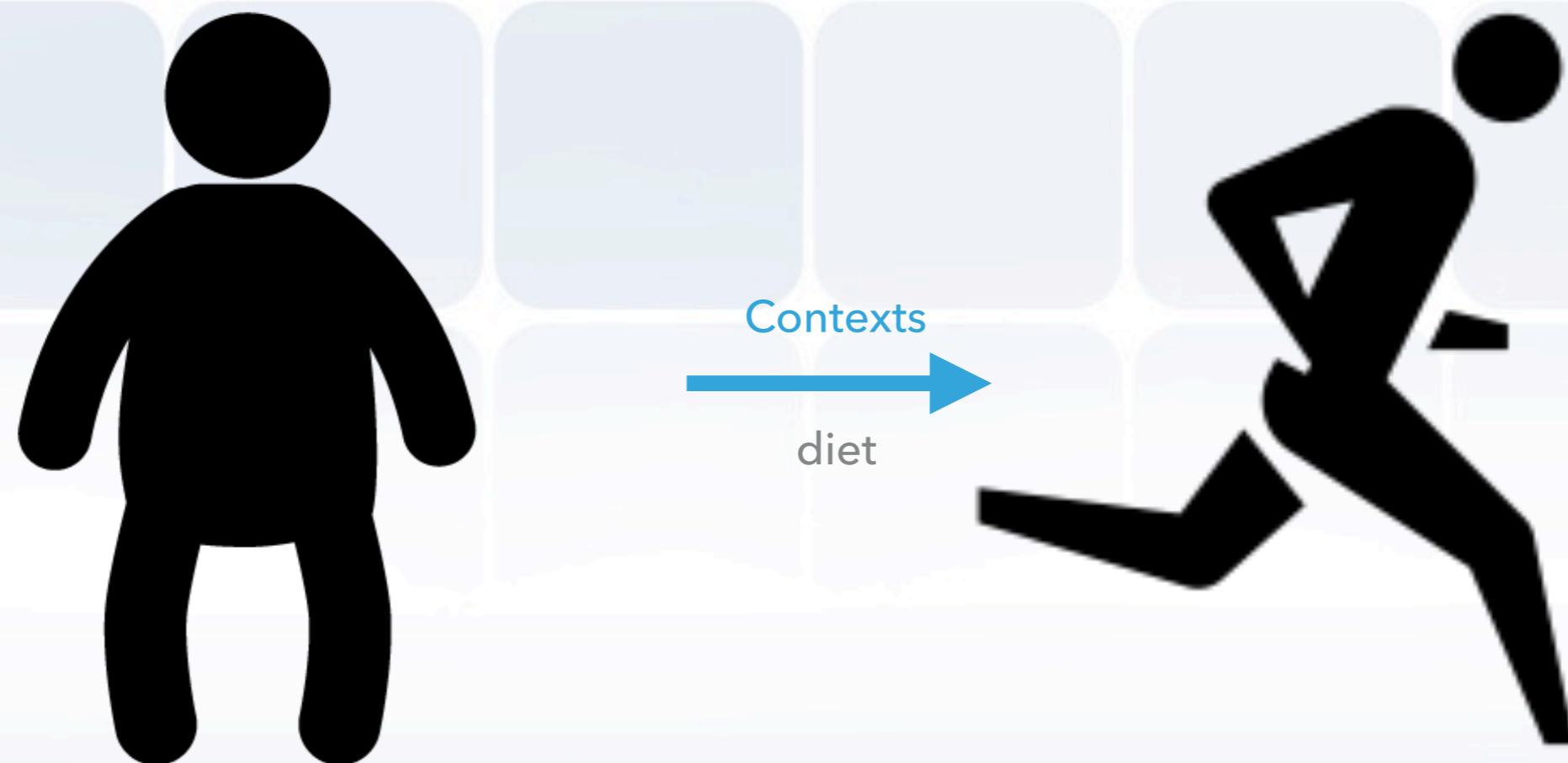
```
books_controller.rb
```

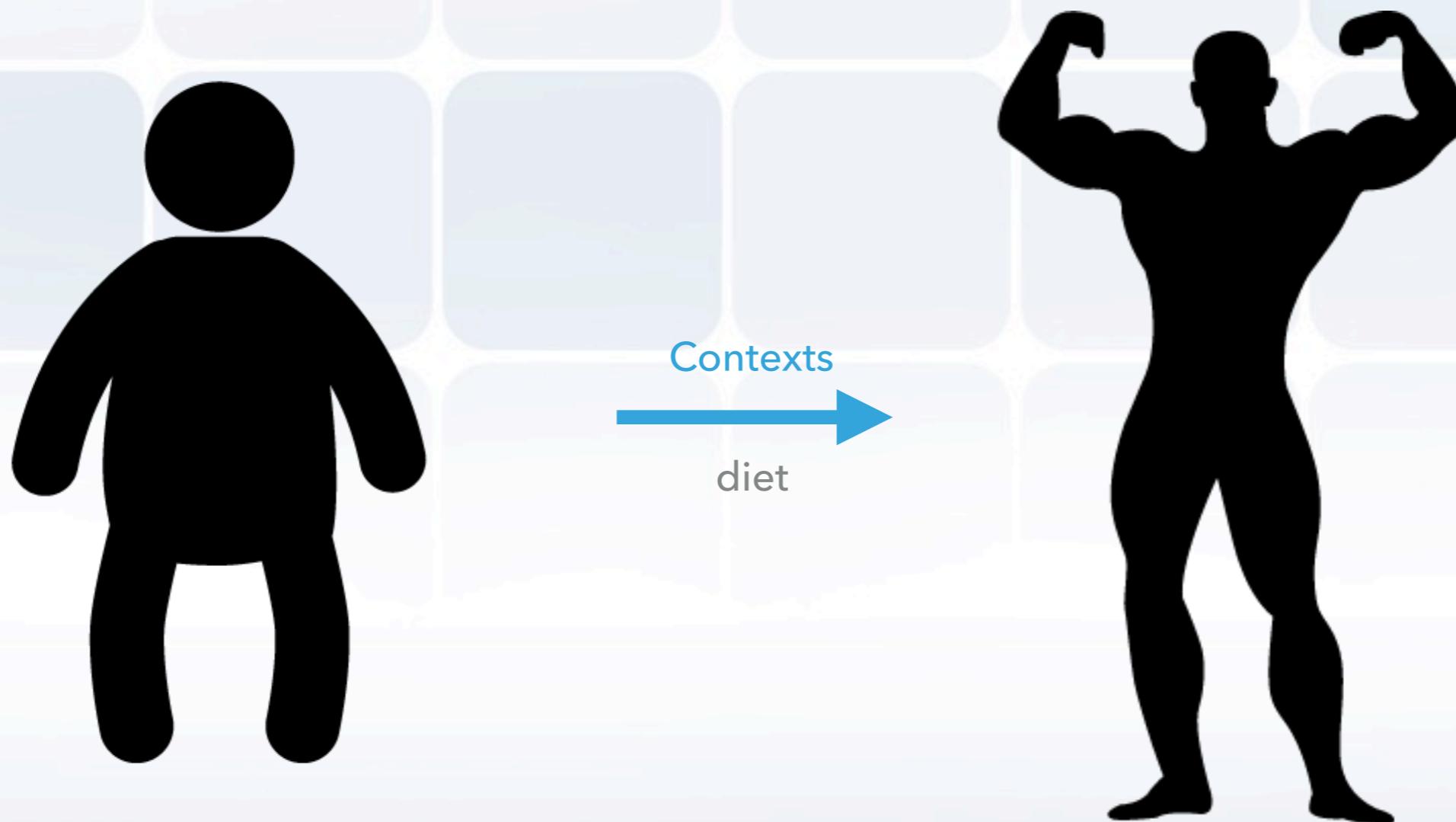
```
books_controller.rb
```

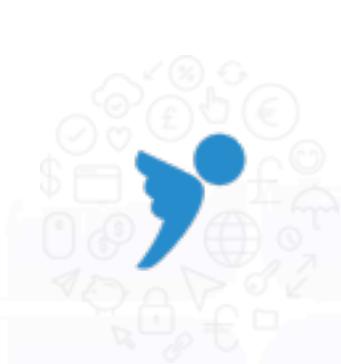
```
1 class BooksController < ActionController::Base
2   before_action :load_context
3   ...
4
5   def edit; end
6
7   def update
8     @context.updater.perform ? redirect_to(book_path(@context.book)) : render(@context.partial_path)
9   end
10
11  private
12
13  def load_context
14    @context = ::BookContext.new(user: current_user, params: params)
15  end
16 end
17
```

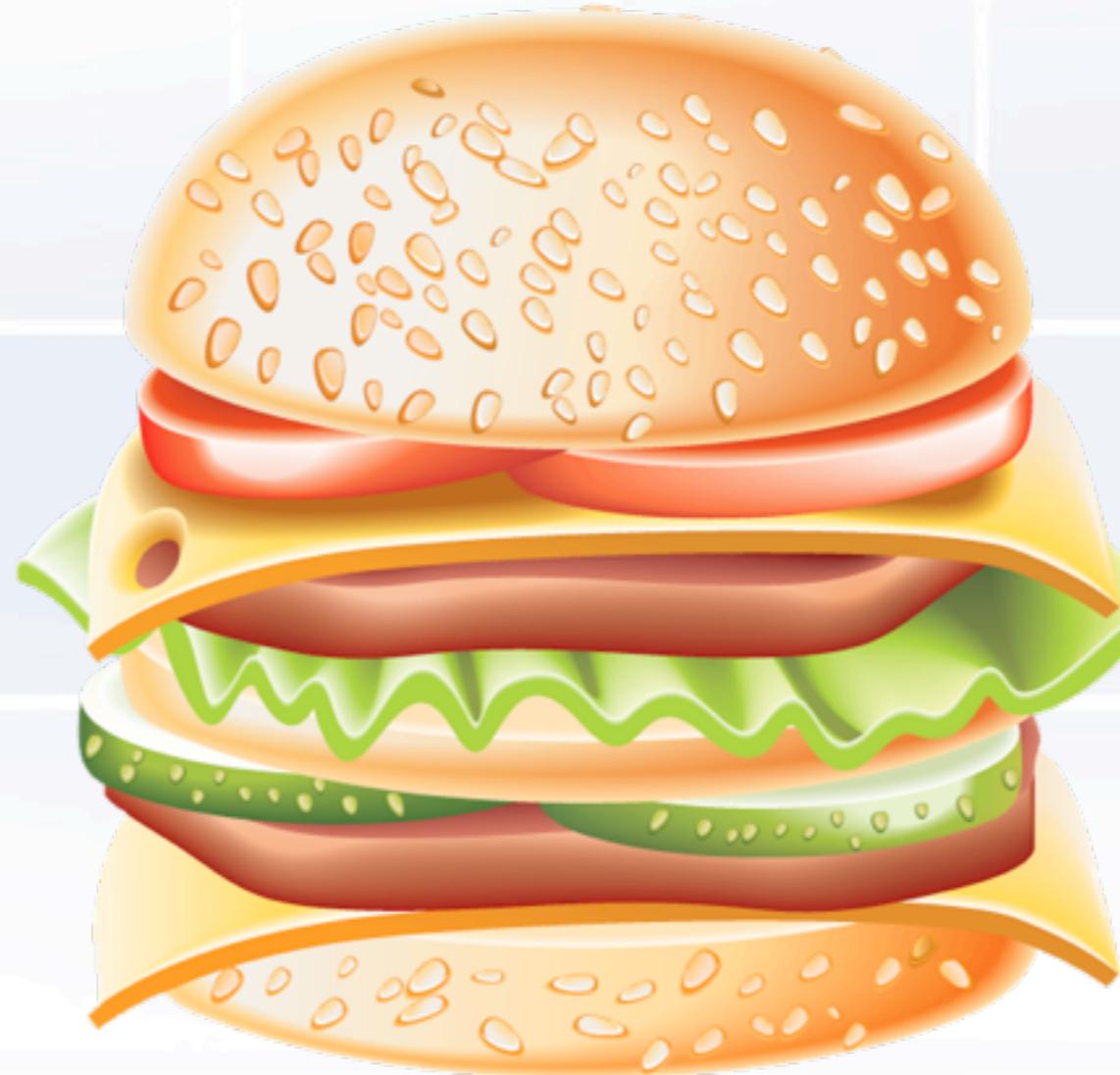
316 LINES
almost 50 %

Line 17, Column 1 Spaces: 2 Ruby







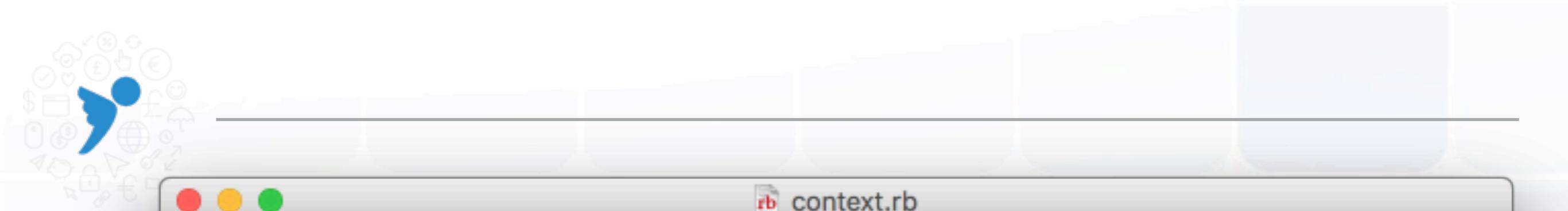


Just a **FAST, slight change.**

5 minutes of work.

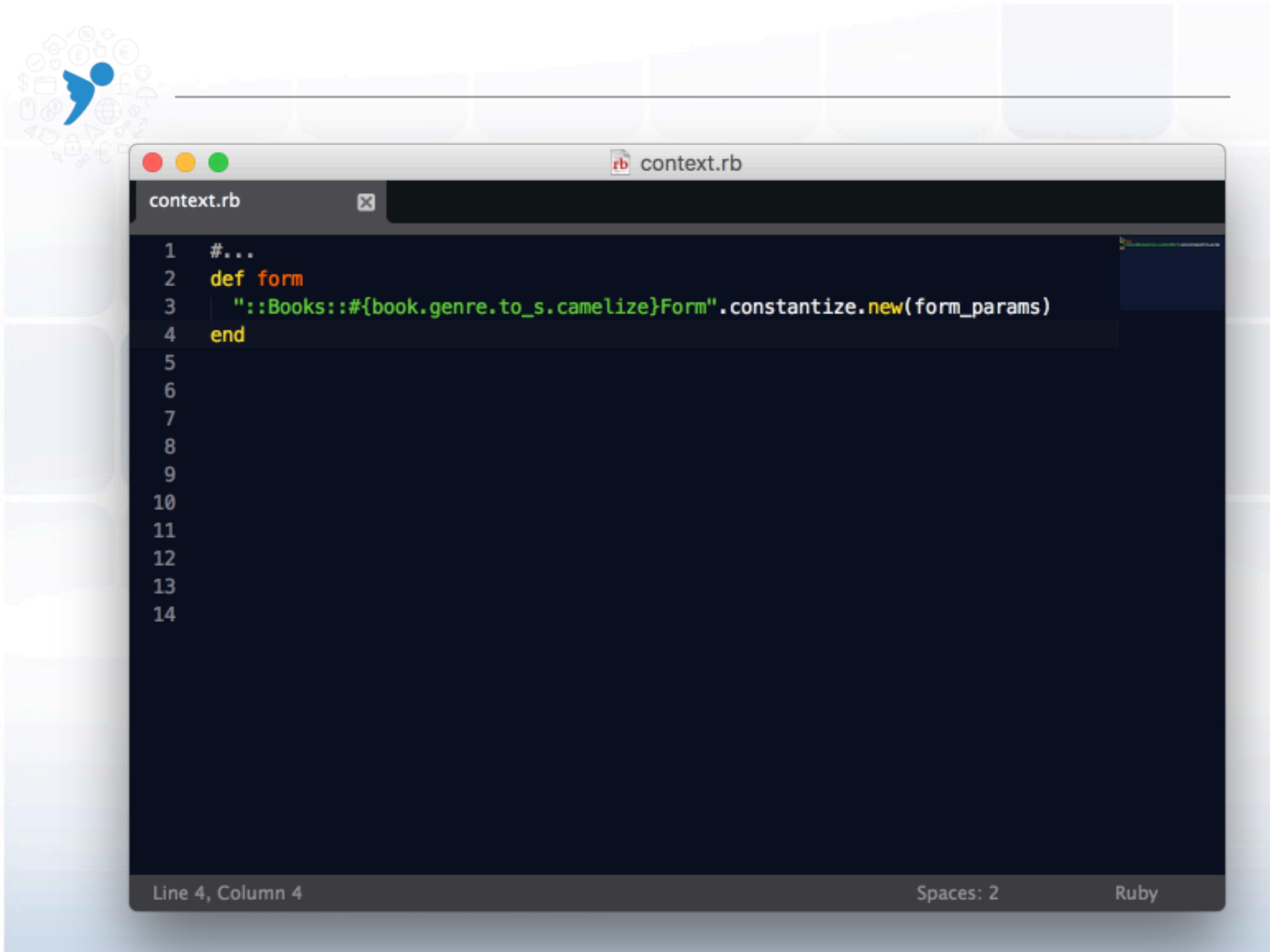


FORMULARZ KSIĄŻKI ZALEŻNY OD JEJ GATUNKU



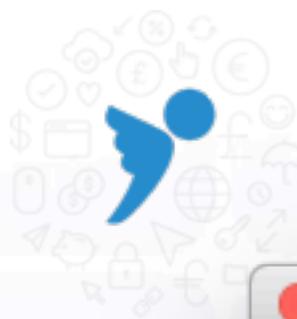
```
context.rb
```

```
1 #...
2 def form
3   user.admin? ? ::Books::AdminForm.new(form_params) : ::Books::UserForm.new(
4     form_params)
5 end
```



```
context.rb
```

```
1 #...
2 def form
3   "::Books::#{book.genre.to_s.camelize}Form".constantize.new(form_params)
4 end
5
6
7
8
9
10
11
12
13
14
```

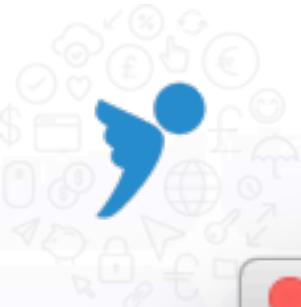


```
context.rb
```

```
1 #...
2 def form
3   case book.genre
4   when :thriller, :horror, :crime, :coaching
5     ::Books::FearForm.new(form_params)
6   when /fiction/, /histor/
7     "::Books::#{book.genre.to_s.camelize}Form".constantize.new(form_params)
8   else
9     ::Books::BasicForm.new(form_params)
10  end
11 end
12
```

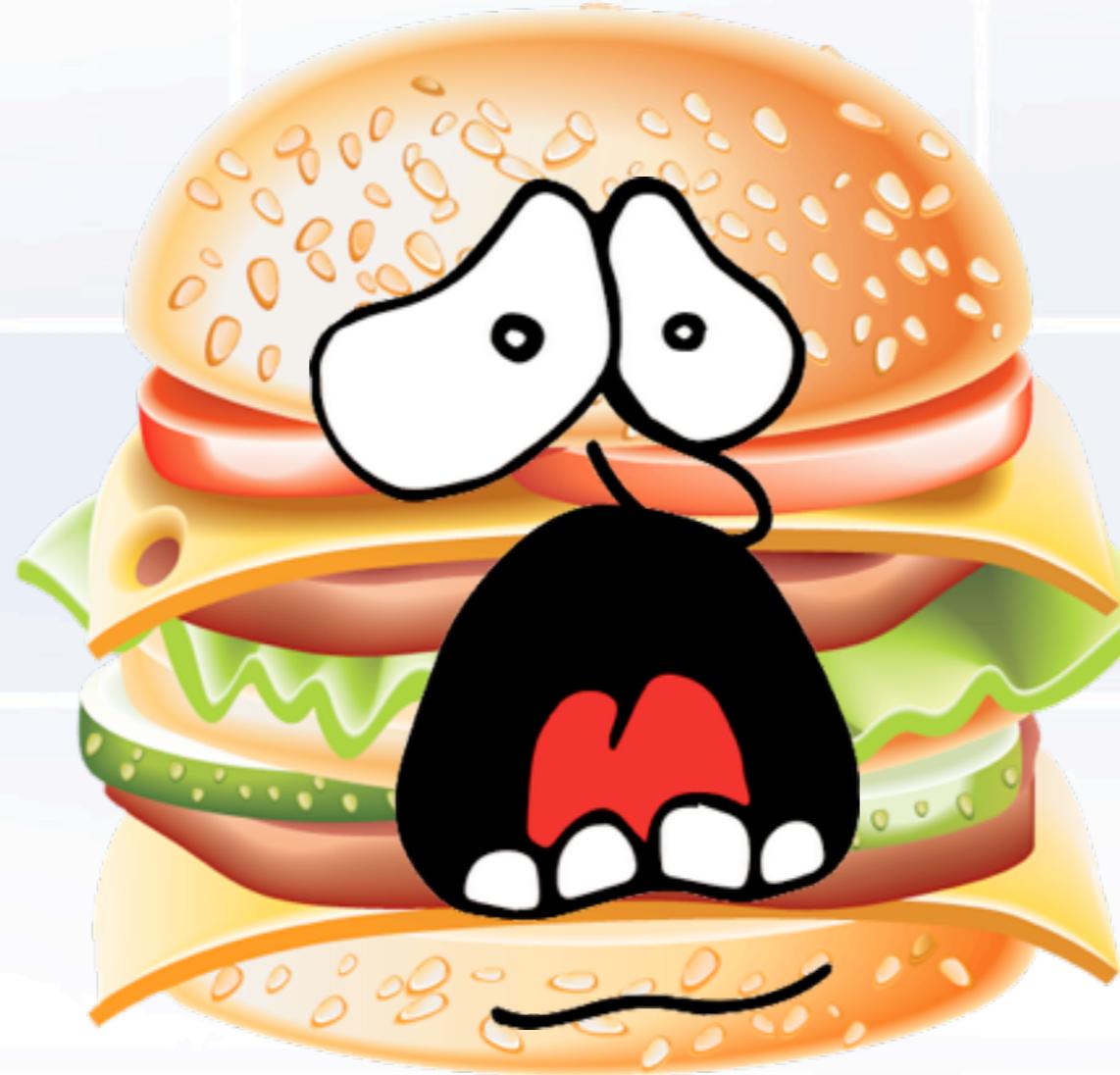
NIEKTÓRE GATUNKI WSPÓLNY NIEKTÓRE WŁASNY POZOSTAŁE BASIC

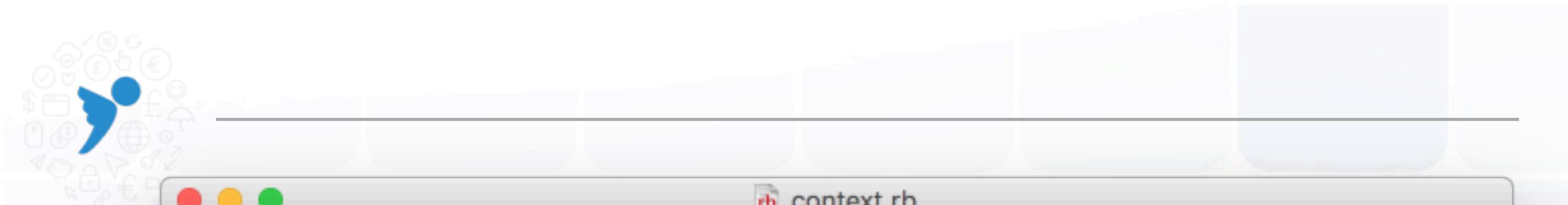
Line 1, Column 1 Spaces: 2 Ruby



```
context.rb
```

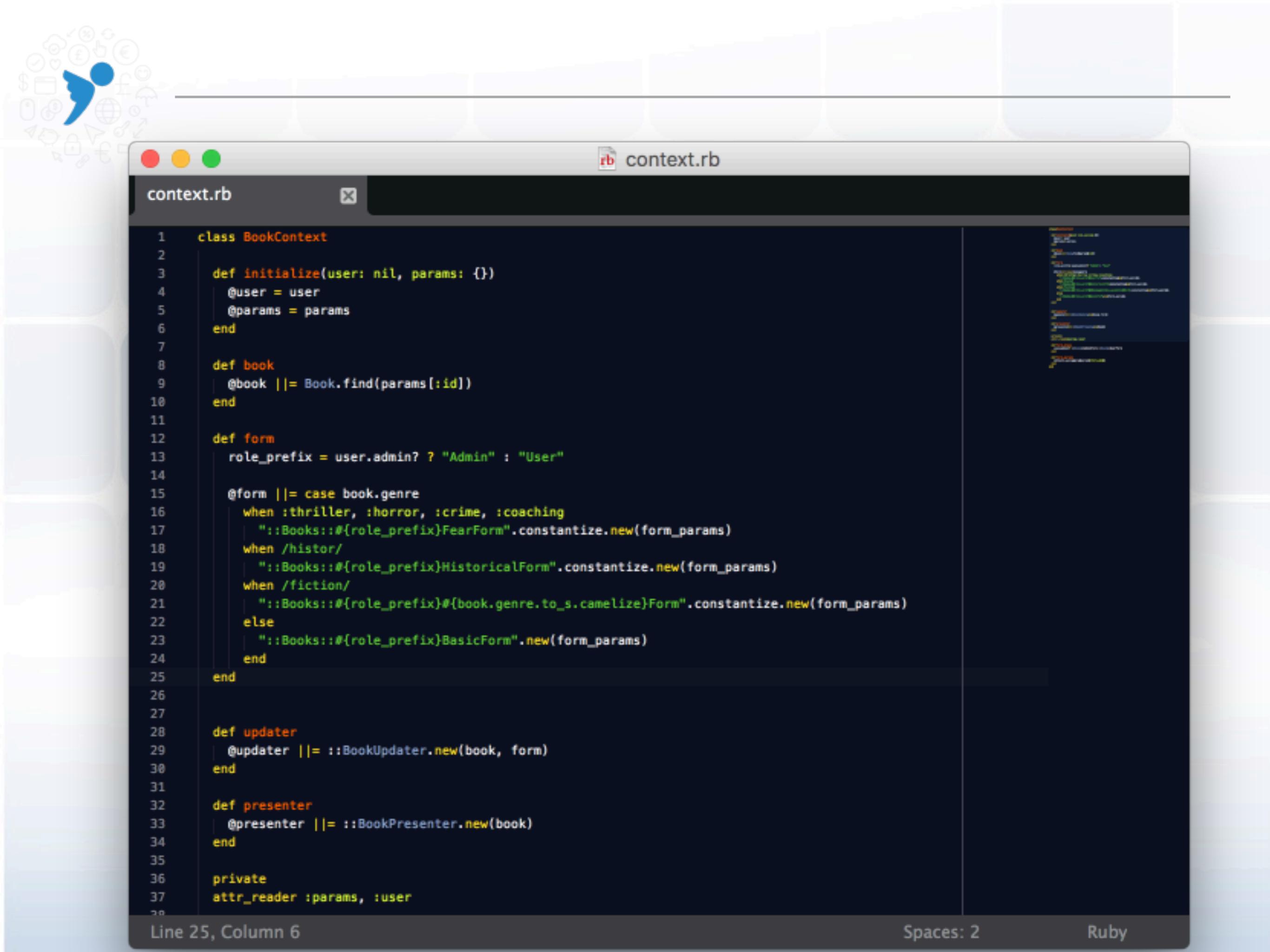
```
1 #...
2 def form
3   case book.genre
4   when :thriller, :horror, :crime, :coaching
5     ::Books::FearForm.new(form_params)
6   when /histor/
7     ::Books::HistoricalForm.new(form_params)
8   when /fiction/
9     "::Books::#{book.genre.to_s.camelize}Form".constantize.new(form_params)
10 else
11   ::Books::BasicForm.new(form_params)
12 end
13 end
14
```





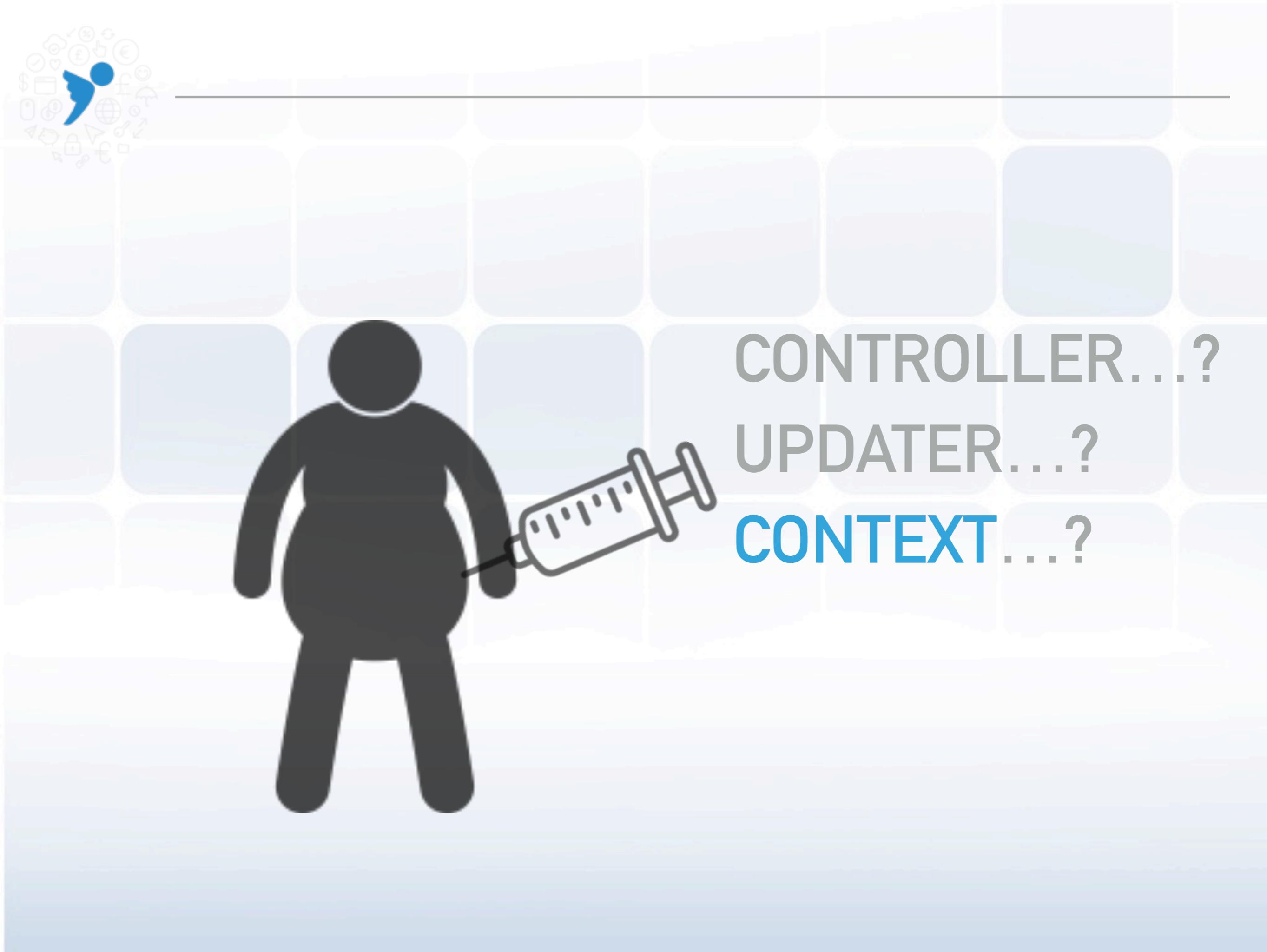
```
context.rb
```

```
1 #...
2 def form
3   role_prefix = user.admin? ? "Admin" : "User"
4
5   case book.genre
6   when :thriller, :horror, :crime, :coaching
7     "::Books::#{role_prefix}FearForm".constantize.new(form_params)
8   when /histor/
9     "::Books::#{role_prefix}HistoricalForm".constantize.new(form_params)
10  when /fiction/
11    "::Books::#{role_prefix}#{book.genre.to_s.camelize}Form".constantize.new(
12      form_params)
13  else
14    "::Books::#{role_prefix}BasicForm".new(form_params)
15  end
16
```



A screenshot of a code editor showing a Ruby file named `context.rb`. The code defines a class `BookContext` with methods for initializing, finding a book, creating a form, updating, and presenting it.

```
1  class BookContext
2
3    def initialize(user: nil, params: {})
4      @user = user
5      @params = params
6    end
7
8    def book
9      @book ||= Book.find(params[:id])
10   end
11
12   def form
13     role_prefix = user.admin? ? "Admin" : "User"
14
15     @form ||= case book.genre
16       when :thriller, :horror, :crime, :coaching
17         "::Books::#{role_prefix}FearForm".constantize.new(form_params)
18       when /histor/
19         "::Books::#{role_prefix}HistoricalForm".constantize.new(form_params)
20       when /fiction/
21         "::Books::#{role_prefix}#{book.genre.to_s.camelize}Form".constantize.new(form_params)
22       else
23         "::Books::#{role_prefix}BasicForm".new(form_params)
24     end
25   end
26
27
28   def updater
29     @updater ||= ::BookUpdater.new(book, form)
30   end
31
32   def presenter
33     @presenter ||= ::BookPresenter.new(book)
34   end
35
36   private
37   attr_reader :params, :user
38
```



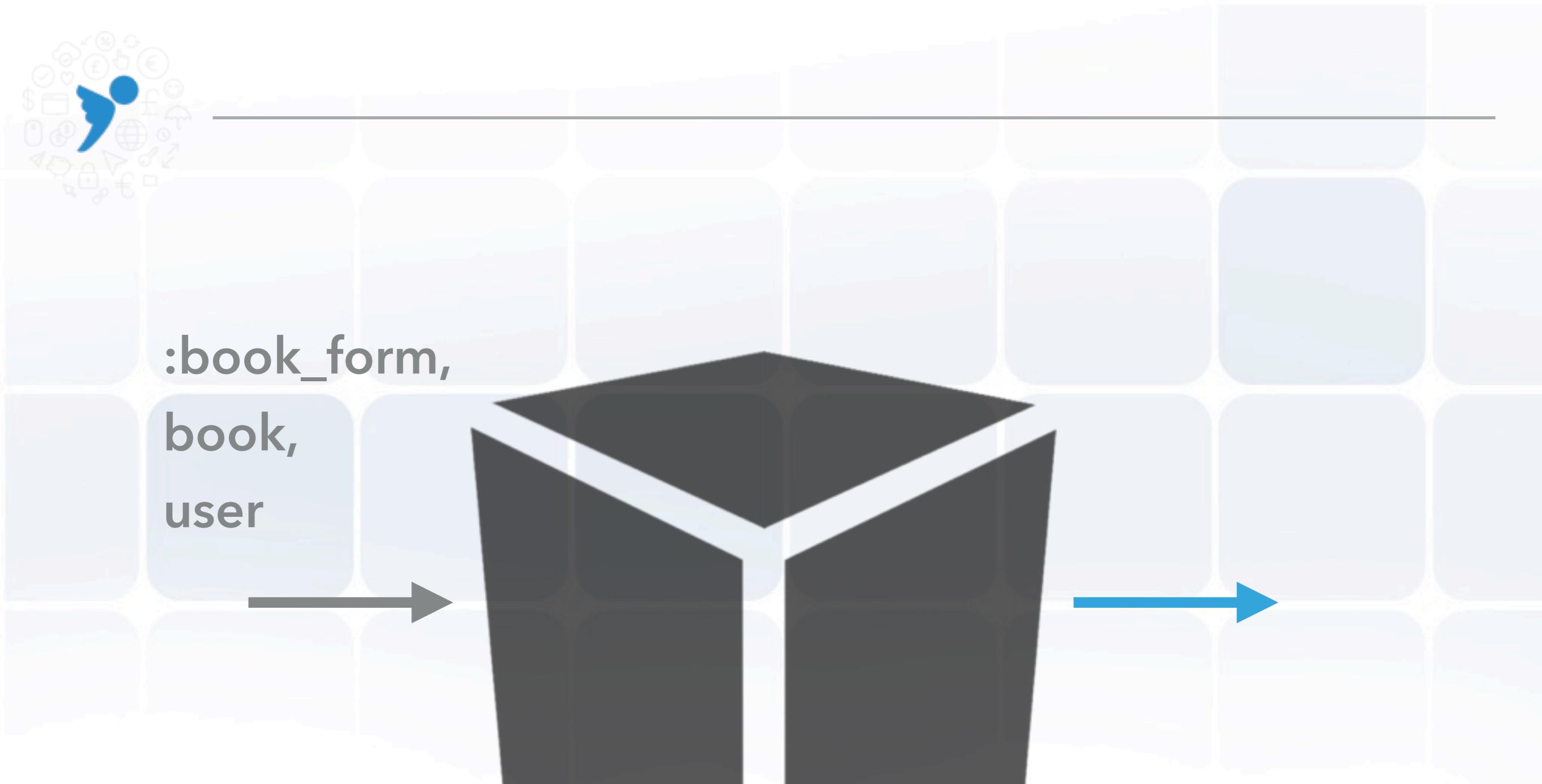
CONTROLLER...?

UPDATER...?

CONTEXT...?



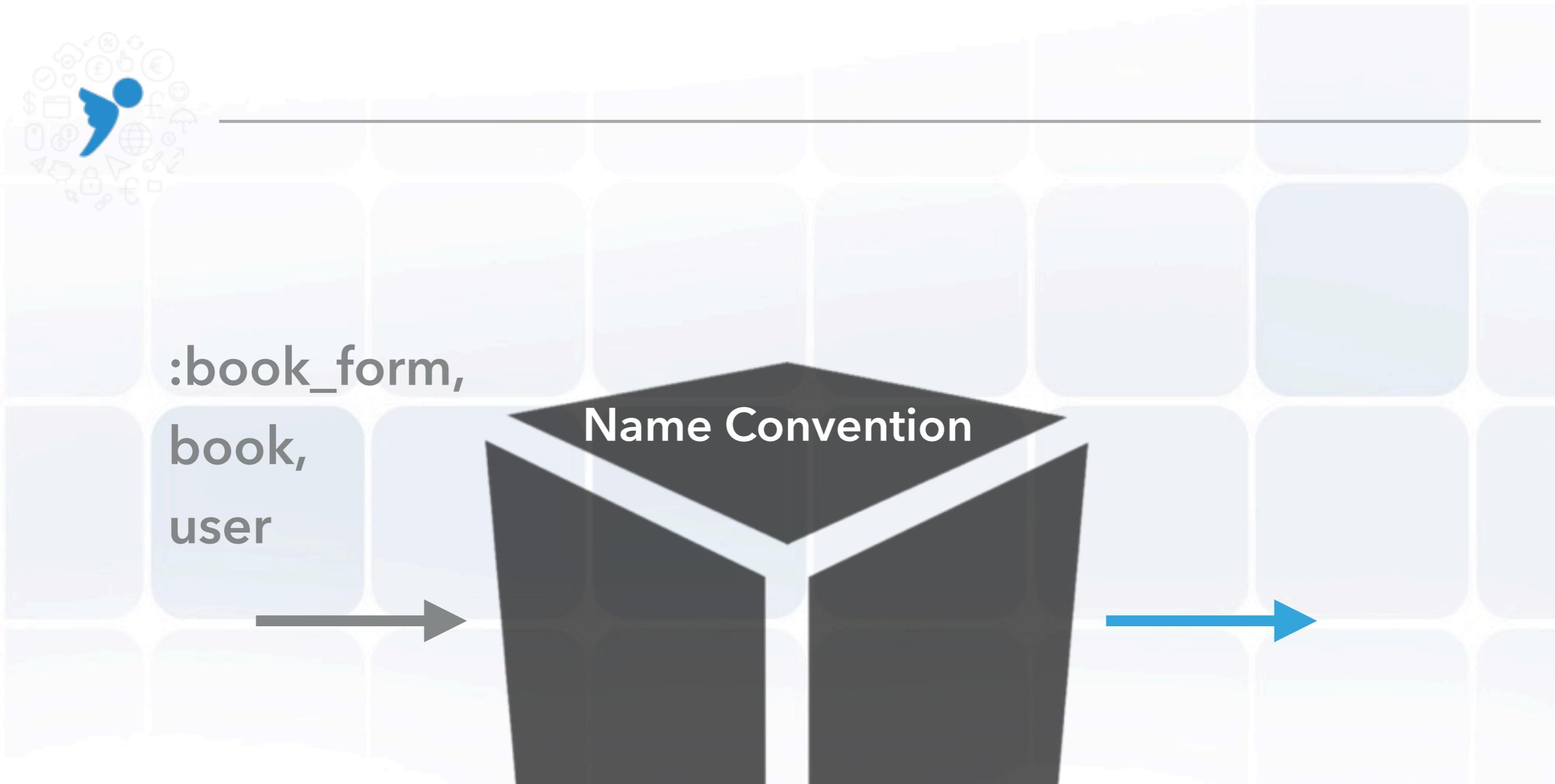
SKINNY CONTROLLERS.
SKINNY MODELS.
...FAT CONTEXTS?



::Book::AdminFearForm

::Book::UserBasicForm

::Book::UserScienceFictionForm



Name Convention

`:book_form`,
`book`,
`user`

`::Book::AdminFearForm`
`::Book::UserBasicForm`
`::Book::UserScienceFictionForm`



SERVICES OBJECTS

SLIMMING THERAPY

CONTEXTS

DIET



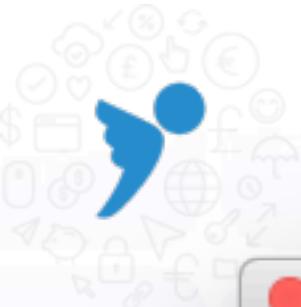
NAME CONVENTION

PERSONAL TRAINER



NAME CONVENTION

DIAGNOZOWANIE KLASY
W ZALEŻNOŚCI
OD KONTEKSTU



```
context.rb
```

```
1 #...
2 def form
3   role_prefix = user.admin? ? "Admin" : "User"
4
5   case book.genre
6   when :thriller, :horror, :crime, :coaching
7     "::Books::#{role_prefix}FearForm".constantize.new(form_params)
8   when /histor/
9     "::Books::#{role_prefix}HistoricalForm".constantize.new(form_params)
10  when /fiction/
11    "::Books::#{role_prefix}#{book.genre.to_s.camelize}Form".constantize.new(
12      form_params)
13  else
14    "::Books::#{role_prefix}BasicForm".new(form_params)
15  end
16
```

Line 16, Column 1 Spaces: 2 Ruby



```
book_form.rb
```

```
1 module NameConvention
2   class BookForm
3     def initialize(user, book)
4       @user = user
5       @book = book
6     end
7
8     def klass_name
9       "::Books::#{role_prefix}#{klass_sufix}Form".constantize
10    end
11
12   private
13   attr_reader :user, :book
14
15   def role_prefix
16     @role_prefix ||= user.admin? ? "Admin" : "User"
17   end
18
19   def klass_sufix
20     case book.genre
21     when :thriller, :horror, :crime, :coaching then "Fear"
22     when /histor/ then "Historical"
23     when /fiction/ then book.genre.to_s.camelize
24     else "Basic"
25   end
26 end
27 end
28 end
```

Line 9, Column 62

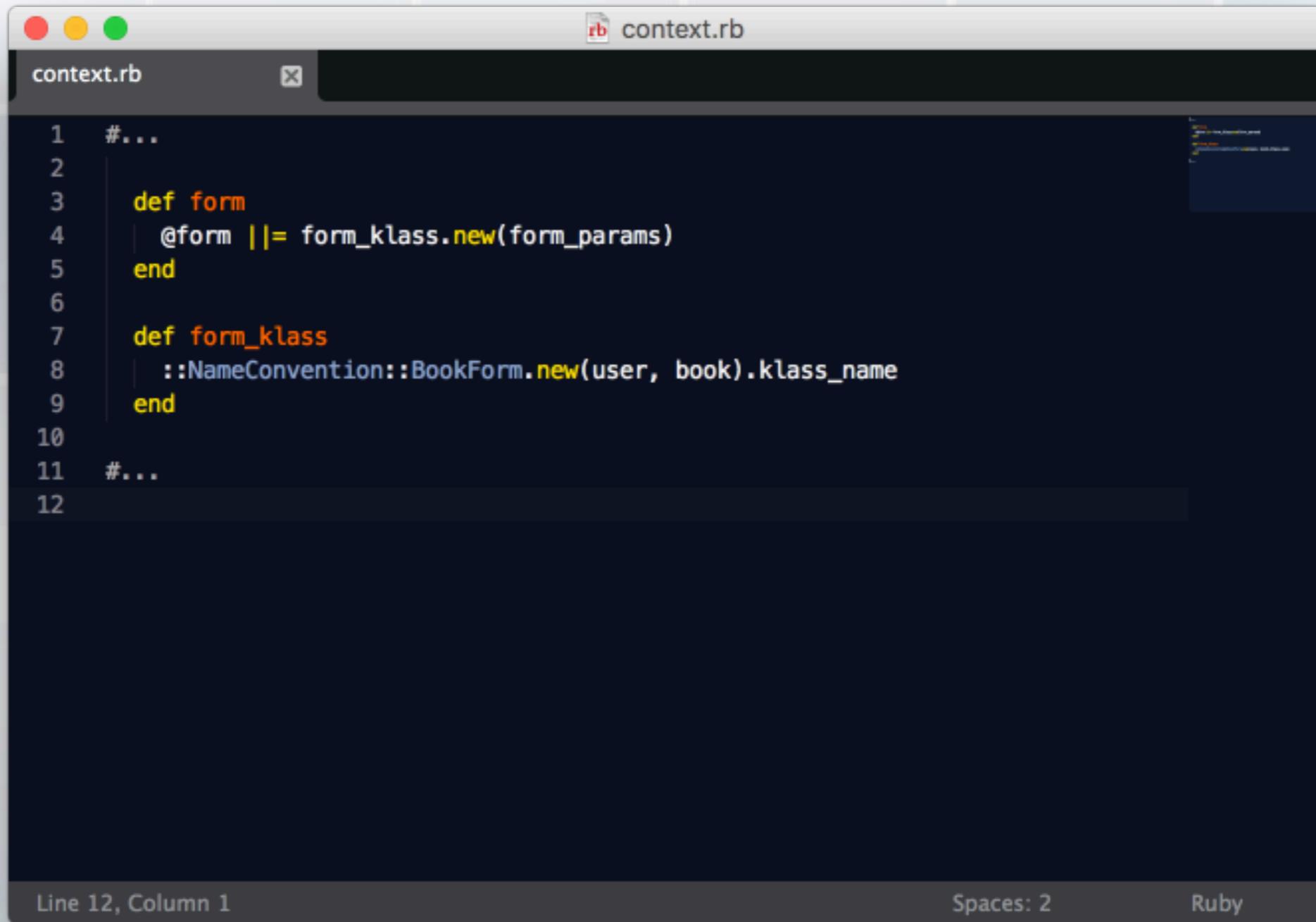
Spaces: 2 Ruby

app

-- services

----- name_convention

----- book_form.rb



```
context.rb
```

```
rb context.rb
```

```
1 #...
2
3 def form
4   @form ||= form_klass.new(form_params)
5 end
6
7 def form_klass
8   ::NameConvention::BookForm.new(user, book).klass_name
9 end
10
11 #...
12
```

Line 12, Column 1 Spaces: 2 Ruby

:book_form,
book,
user



::Book::AdminFearForm
::Book::UserBasicForm
::Book::UserScienceFictionForm

NameConvention.klass_name(:**book_form**, user: user, book: book)

NameConvention.klass_name(:**book_updater**, book)

```
name_convention.rb
```

```
1 module NameConvention
2   module_function
3
4   def identify
5   end
6
7   def klass_name
8   end
9 end
```

10

app

--- services

----- book_updater.rb

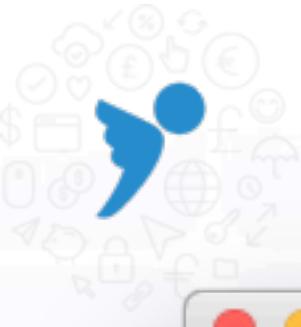
----- book_context.rb

----- name_convention.rb

Line 10, Column 1

Spaces: 2

Ruby

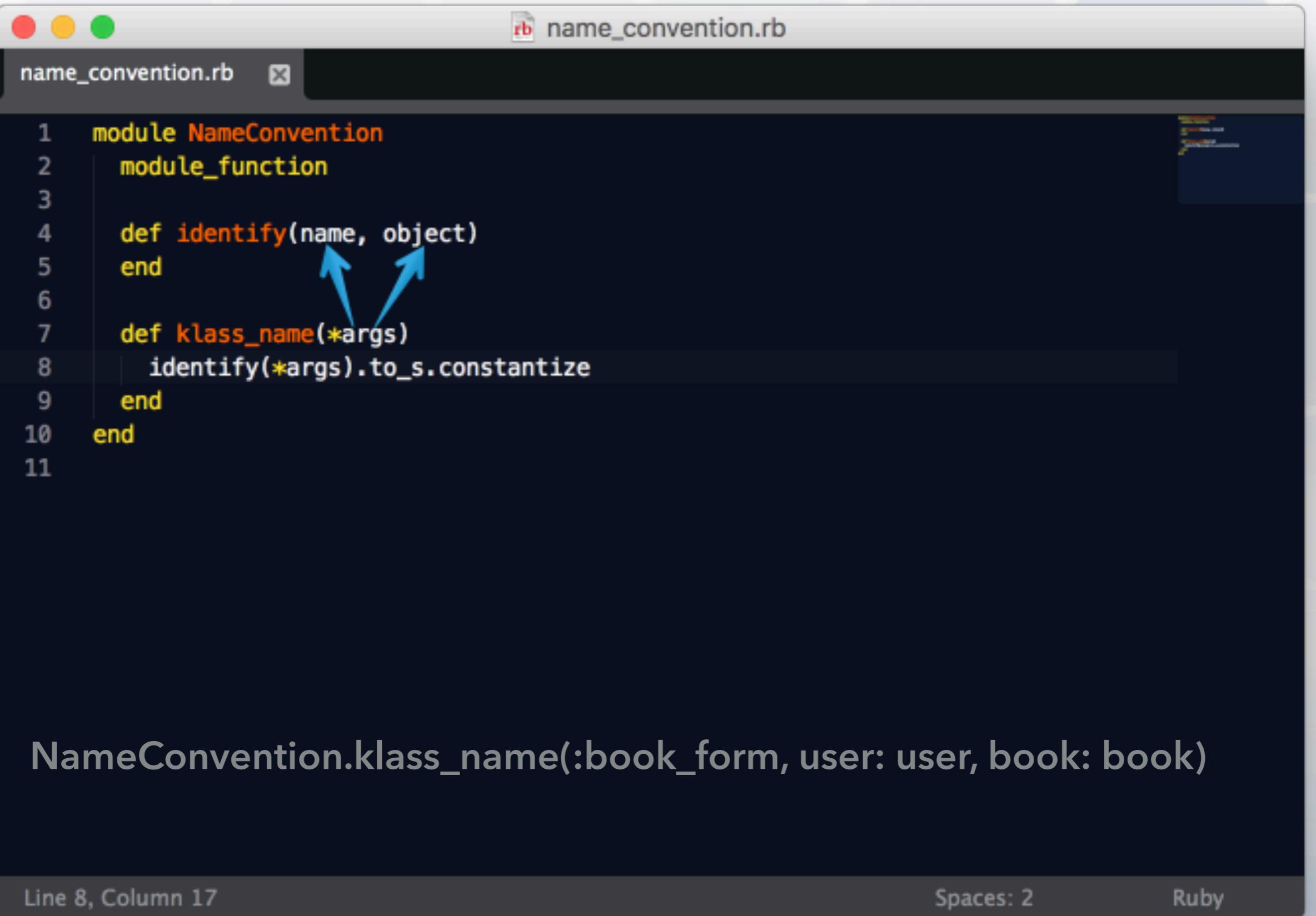


```
name_convention.rb
```

```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     end
6
7   def klass_name(*args)
8     identify(*args).to_s.constantize
9   end
10  end
11
```

NameConvention.klass_name(:book_form, user: user, book: book)

Line 8, Column 37 Spaces: 2 Ruby



A screenshot of a code editor window titled "name_convention.rb". The code defines a module "NameConvention" with a nested module "function". It contains two methods: "identify" which takes "name" and "object" parameters, and "klass_name" which takes multiple arguments ("*args") and returns the result of "identify(*args).to_s.constantize". Two blue arrows point from the text "NameConvention" in the title bar to the "identify" method's first parameter "name".

```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     end
6
7   def klass_name(*args)
8     identify(*args).to_s.constantize
9   end
10 end
11
```

NameConvention.klass_name(:book_form, user: user, book: book)

Line 8, Column 17 Spaces: 2 Ruby



```
name_convention.rb
```

```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     "NameConvention::#{name.to_s.camelize}"
6   end
7
8   def klass_name(*args)
9     identify(*args).to_s.constantize
10  end
11 end
12
```

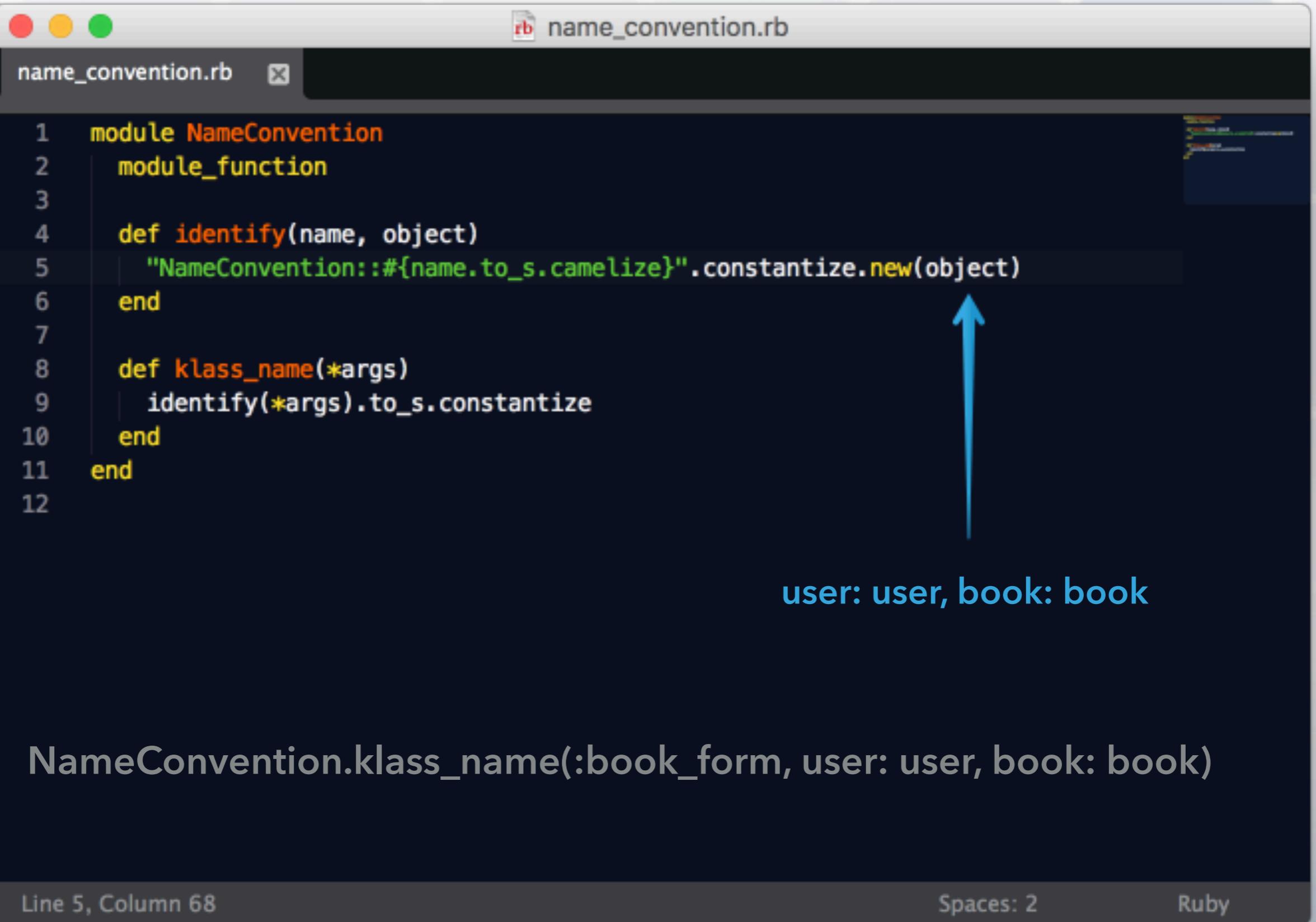
NameConvention.klass_name(:book_form, user: user, book: book)



```
name_convention.rb
```

```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     "NameConvention::#{name.to_s.camelize}"  "NameConvention::BookForm"
6   end
7
8   def klass_name(*args)
9     identify(*args).to_s.constantize
10  end
11 end
12
```

NameConvention.klass_name(:book_form, user: user, book: book)



```
name_convention.rb
```

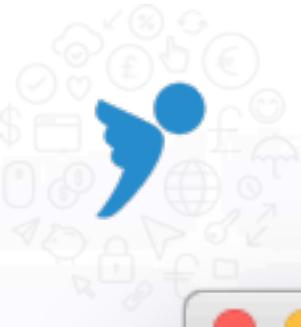
```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     "NameConvention::#{name.to_s.camelize}".constantize.new(object)
6   end
7
8   def klass_name(*args)
9     identify(*args).to_s.constantize
10  end
11 end
12
```

A blue arrow points from the text "user: user, book: book" to the line "constantize.new(object)" in the code.

user: user, book: book

NameConvention.klass_name(:book_form, user: user, book: book)

Line 5, Column 68 Spaces: 2 Ruby



```
name_convention.rb
```

```
1 module NameConvention
2   module_function
3
4   def identify(name, object)
5     "NameConvention::#{name.to_s.camelize}".constantize.new(object).name
6   end
7
8   def klass_name(*args)
9     identify(*args).to_s.constantize
10  end
11 end
12
```

NameConvention.klass_name(:book_form, user: user, book: book)

Line 5, Column 73 Spaces: 2 Ruby



book_form.rb

```
1 module NameConvention
2   class BookForm
3     def initialize(user, book) ←
4       @user = user
5       @book = book
6     end
7
8     def klass_name
9       "::Books::#{role_prefix}#{klass_sufix}Form".constantize ←
10    end
11
12   private
13   attr_reader :user, :book
14
15   def role_prefix
16     @role_prefix ||= user.admin? ? "Admin" : "User"
17   end
18
19   def klass_sufix
20     case book.genre
21     when :thriller, :horror, :crime, :coaching then "Fear"
22     when /histor/ then "Historical"
23     when /fiction/ then book.genre.to_s.camelize
24     else "Basic"
25   end
26 end
27 end
28 end
```

Line 9, Column 62 Spaces: 2 Ruby



book_form.rb

```
1 module NameConvention
2   class BookForm
3     def initialize(user: nil, book: nil)
4       @user = user
5       @book = book
6     end
7
8     def name
9       "::Books::#{role_prefix}#{klass_sufix}Form"
10    end
11
12    private
13    attr_reader :user, :book
14
15    def role_prefix
16      @role_prefix ||= user.admin? ? "Admin" : "User"
17    end
18
19    def klass_sufix
20      case book.genre
21      when :thriller, :horror, :crime, :coaching then "Fear"
22      when /histor/ then "Historical"
23      when /fiction/ then book.genre.to_s.camelize
24      else "Basic"
25      end
26    end
27  end
28 end
```

Line 9, Column 50

Spaces: 2

Ruby

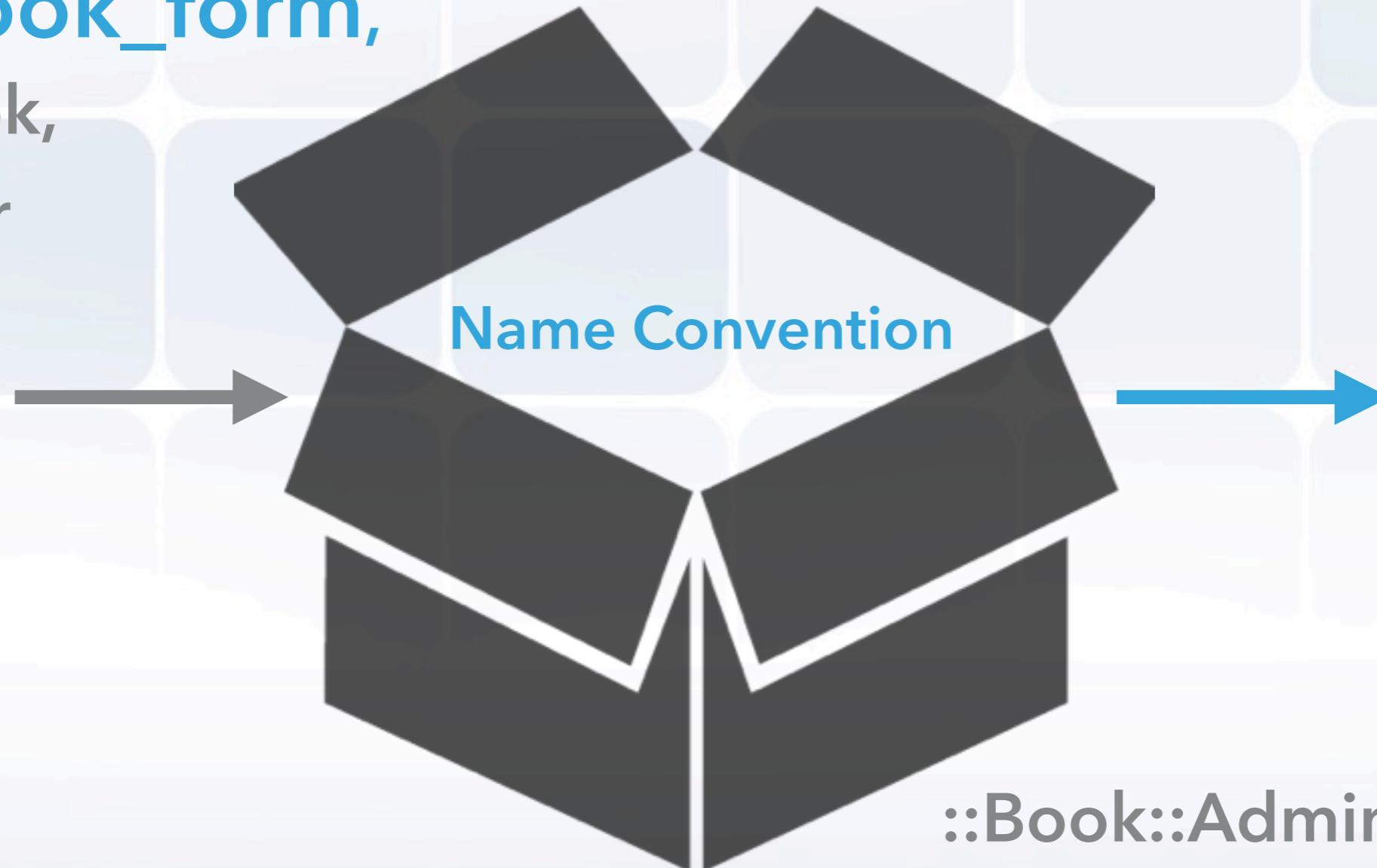


```
context.rb
```

```
1 #...
2
3 def form
4   @form ||= form_klass.new(form_params)
5 end
6
7 def form_klass
8   ::NameConvention.klass_name(:book_form, user: user, book: book)
9 end
10
11 #...
12
```

Line 12, Column 1 Spaces: 2 Ruby

:book_form,
book,
user



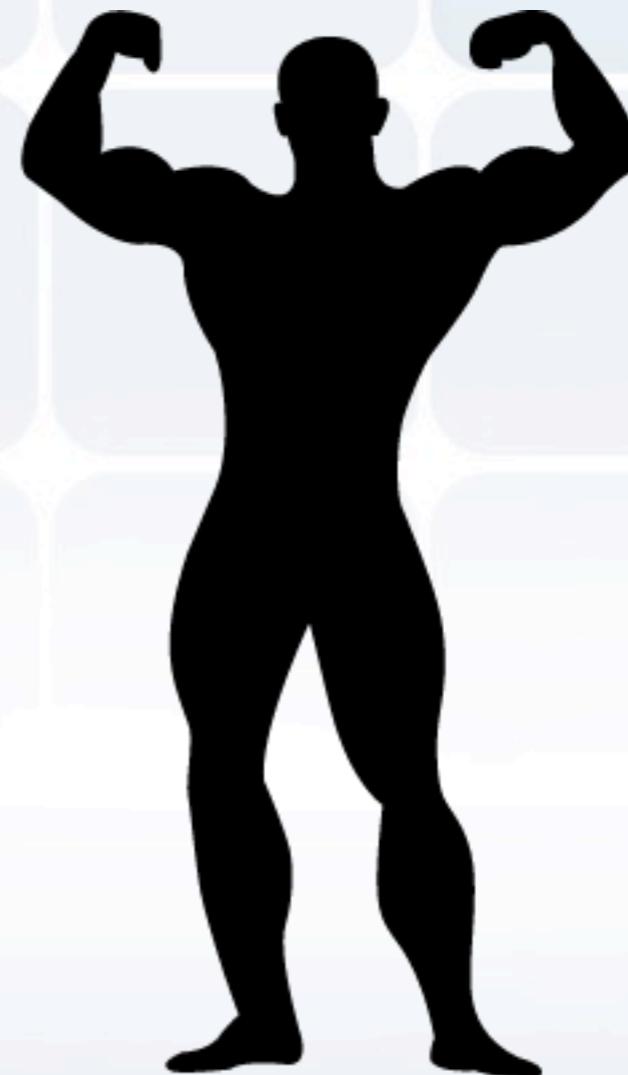
::Book::AdminForm
::Book::UserBasicForm
::Book::UserScienceFictionForm



Name Convention



Personal Trainer







SERVICES OBJECTS

(SKOMPLIKOWANA) LOGIKA
BIZNESOWA PRZENIESIONA
DO SERWISÓW



CONTEXTS



ELEMENTY TWORZĄCE KONTEKST
WYWOŁANIA - DO SERWISÓW
KONTEKSTOWYCH

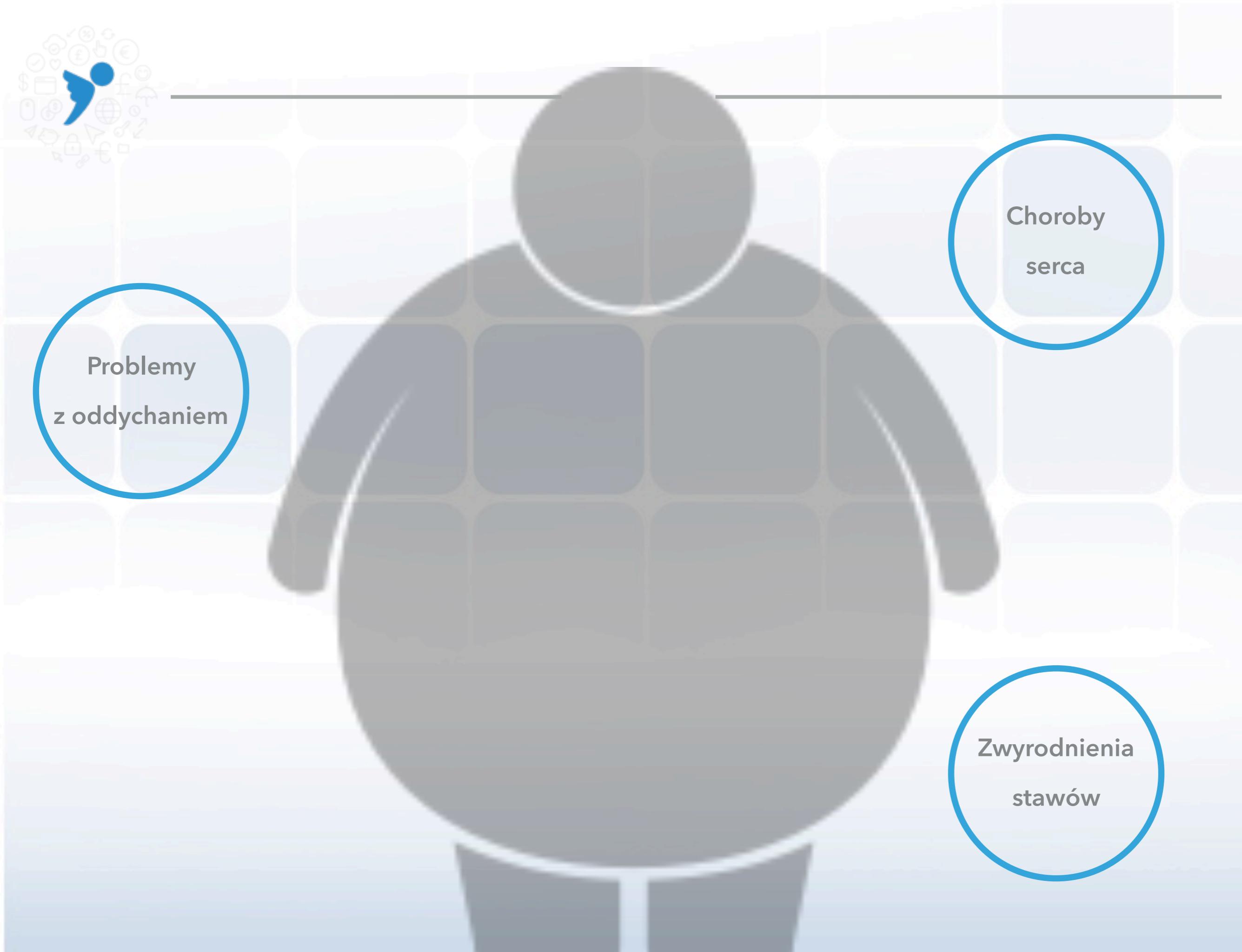


NAME CONVENTION



STWORZONY MECHANIZM
ODCIAŻAJĄCY KOD
Z DIAGNOZOWANIA KLASY





Problemy
z oddychaniem

Choroby
serca

Zwyrodnienia
stawów



Maintainability

Skalowalność

Stabilność

Debuggowanie



**MAKE YOUR CODE
FIT AGAIN**



Alicja Cyganiewicz

alicja.cyganiewicz@infakt.pl

KRUG #6 / 2017

powered by Railsware.pl

17.10.2017 r.



MAKE YOUR CODE
FIT AGAIN

Warsztaty Ruby on Rails z inFakt



www.infakt.pl/warsztaty-rails

www.meetup.com/Warsztaty-Ruby-on-Rails-z-inFakt/

FOR FREE

8, 15, 22 listopada

junior+

5 mentorów