HTML5 and CSS3 for Mobile Applications

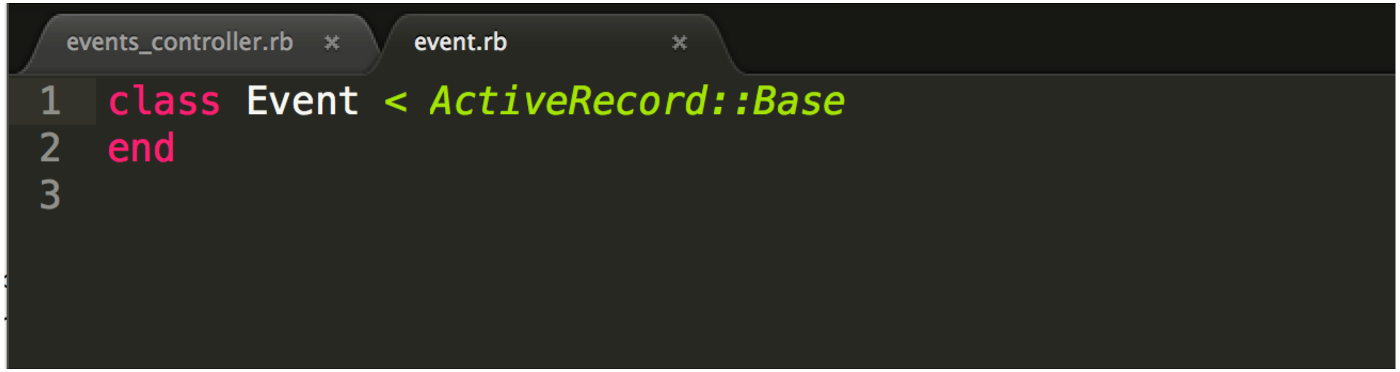
Prof. Paul Krause, University of Surrey

Validations, Action Callbacks and Authentication

Objectives for today

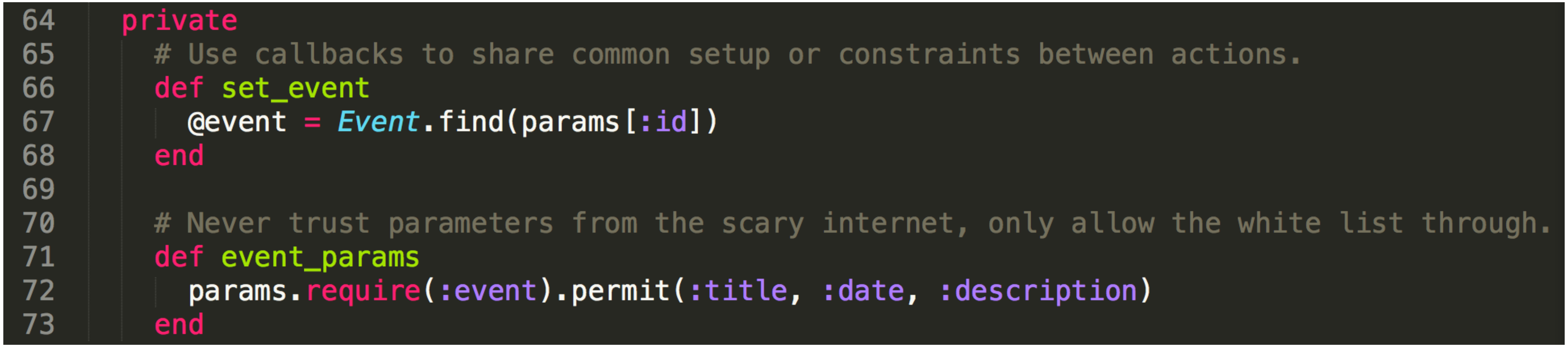
- Add validations to ensure models have all the required attributes
- Show you how to use action callbacks
- Implement an authentication filter using the *Devise* gem

app/models/event.rb



```ruby
class Event < ActiveRecord::Base
end

```

The white-list in app/controllers/events_controller.rb



```ruby
  private
    # Use callbacks to share common setup or constraints between actions.
    def set_event
      @event = Event.find(params[:id])
    end

    # Never trust parameters from the scary internet, only allow the white list through.
    def event_params
      params.require(:event).permit(:title, :date, :description)
    end
```

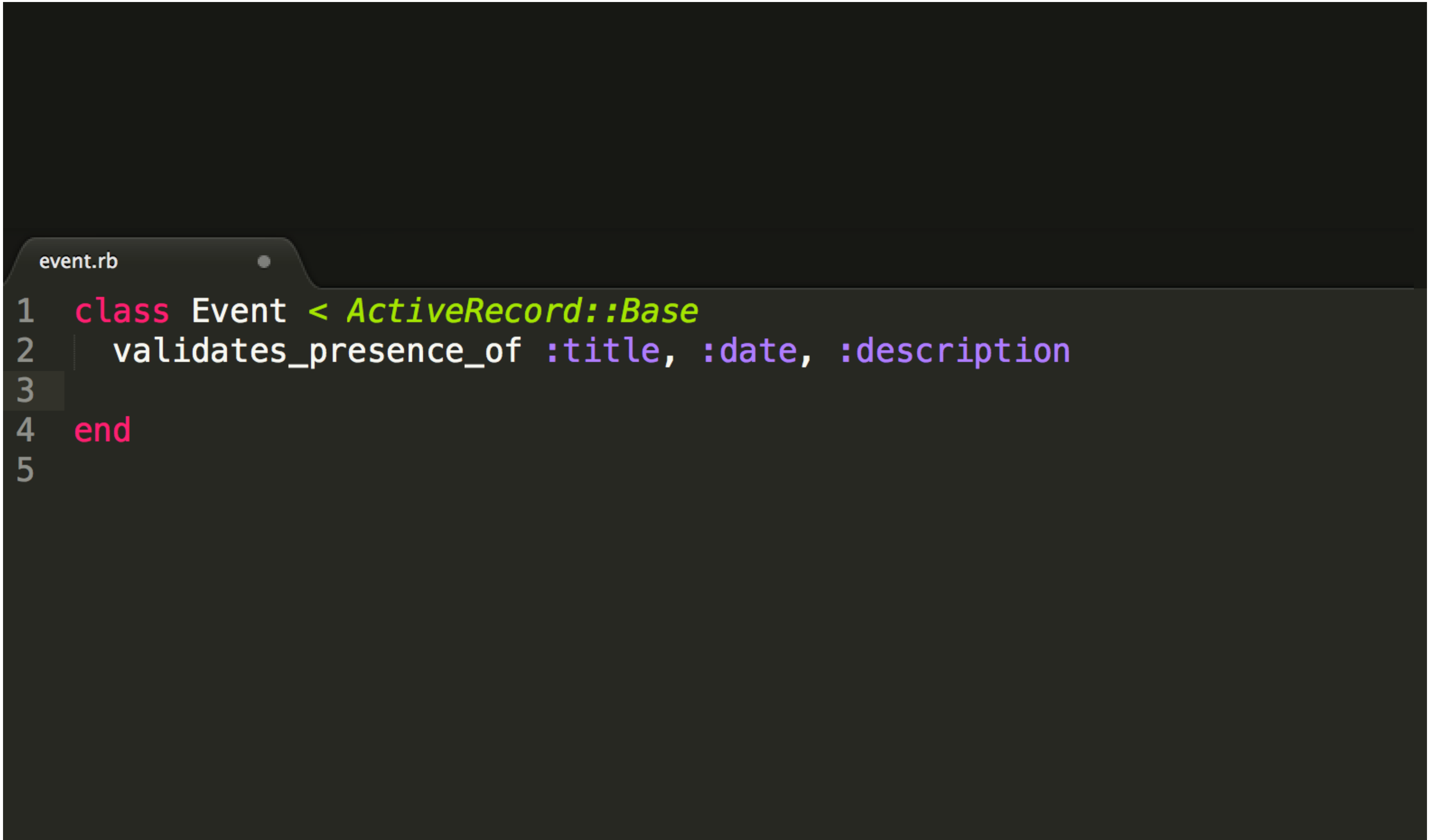
Change “may” to “must”

- The controller white-list identifies the model attributes that *may* be accessed by the controller methods

- However, when creating a new instance or editing details of an existing one there could be attributes that *must* be present

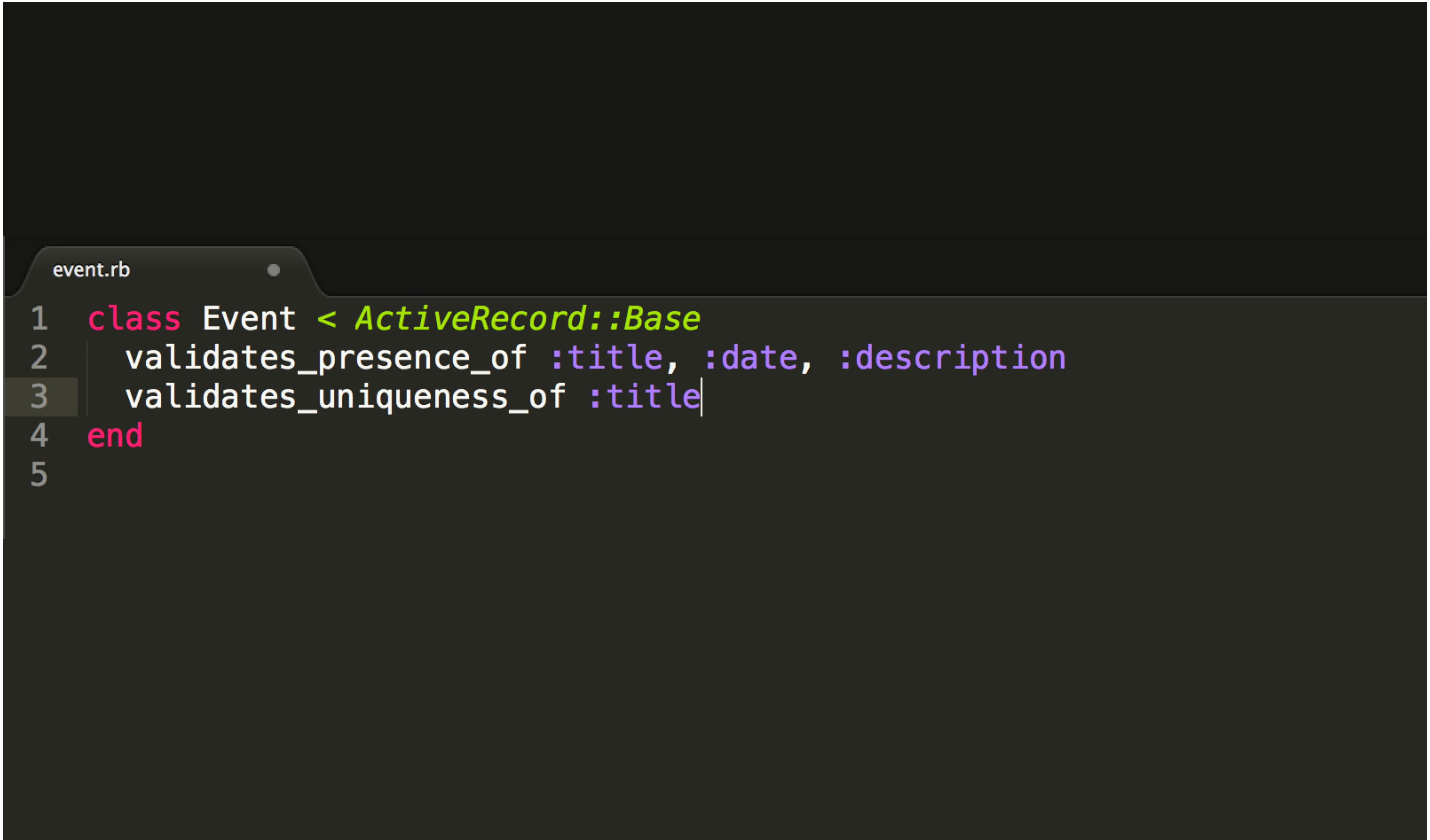
Rails Validations!

app/models/event.rb



```ruby
class Event < ActiveRecord::Base
  validates_presence_of :title, :date, :description

end
```

You might want to add



```ruby
class Event < ActiveRecord::Base
  validates_presence_of :title, :date, :description
  validates_uniqueness_of :title
end
```

Other useful ones:

- *validates_length_of*
  - can specify :minumum => 6; :within => 5 .. 10; :is => 42
- *validates_numericality_of*
  - can specify :equal_to; :greater_than; :less_than; ...
- *validates_with*
  - enables you to use your own customised validator
- For more see: *The Rails 4 Way*, Chapter 8

Action Callbacks & Authorisation

Step 1: Action Callbacks

- We will include an action callback in our Events Controller

  before_action :authenticate_member!

- Make sure your action callback methods are declared private or protected

- You can also use

  - *after_action*

  - *around_action* (use *yield* in its definition)

but, but, but ...

- We don't want to filter all our controller actions

```
before_action :authenticate_member!, :except => [:index, :show]
```

- We could also have used:

```
before_action :authenticate_member!, :only => [:new, :edit, :create, :update, :destroy]
```

but, but, but ...

- We don’t want to filter all our controller actions

  before_action :authenticate_member!, :except => [:index, :show]

- We could also have used:

before_action :authenticate_member!, :only => [:new, :edit, :create, :update, :destroy]

For more on Action Callbacks, see
The Rails 4 Way, Section 4.7

Now let’s sort out the authentication

Health Warning!

- There is an argument for coding up the authorisation yourself first time round
- I disagree!
  - It takes too long for a first course in Rails
- We will use the *devise* gem
  - *https://github.com/plataformatec/devise*

Installing Gems

- Add the gem to your project by including it in your Gemfile
  - You will find this in the root directory of your project
  - Add the line: *gem* ‘devise’
- You then need to install the gem file by running

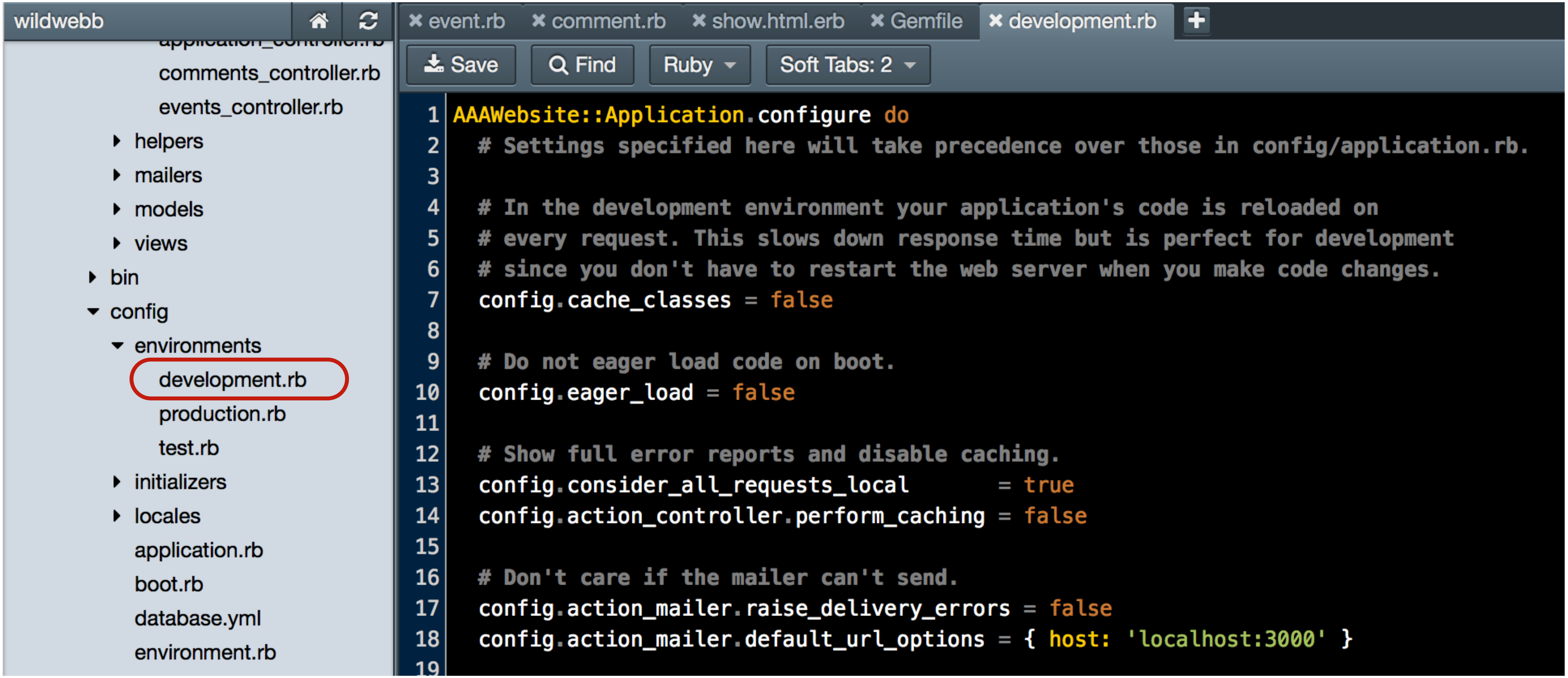
  $ *bundle install*

Setting up *devise*

- *rails generate devise:install*
  - You might need to fix some recommendations at this point

Setting up *devise*

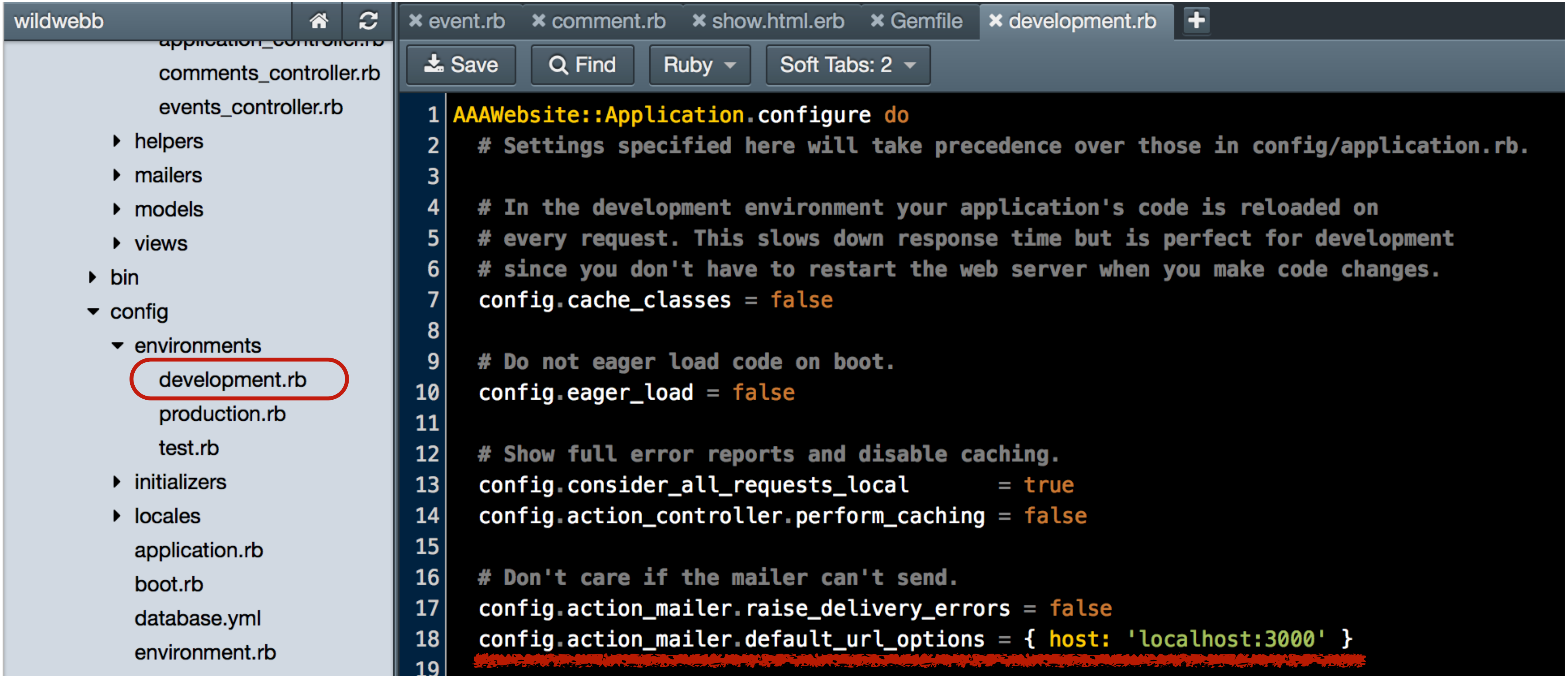
- *rails generate devise:install*
  - You might need to fix some recommendations at this point



```ruby
AAAWebsite::Application.configure do
  # Settings specified here will take precedence over those in config/application.rb.

  # In the development environment your application's code is reloaded on
  # every request. This slows down response time but is perfect for development
  # since you don't have to restart the web server when you make code changes.
  config.cache_classes = false

  # Do not eager load code on boot.
  config.eager_load = false

  # Show full error reports and disable caching.
  config.consider_all_requests_local       = true
  config.action_controller.perform_caching = false

  # Don't care if the mailer can't send.
  config.action_mailer.raise_delivery_errors = false
  config.action_mailer.default_url_options = { host: 'localhost:3000' }
```

Setting up *devise*

- *rails generate devise:install*
  - You might need to fix some recommendations at this point



```ruby
AAAWebsite::Application.configure do
  # Settings specified here will take precedence over those in config/application.rb.

  # In the development environment your application's code is reloaded on
  # every request. This slows down response time but is perfect for development
  # since you don't have to restart the web server when you make code changes.
  config.cache_classes = false

  # Do not eager load code on boot.
  config.eager_load = false

  # Show full error reports and disable caching.
  config.consider_all_requests_local       = true
  config.action_controller.perform_caching = false

  # Don't care if the mailer can't send.
  config.action_mailer.raise_delivery_errors = false
  config.action_mailer.default_url_options = { host: 'localhost:3000' }
```

Setting up *devise*

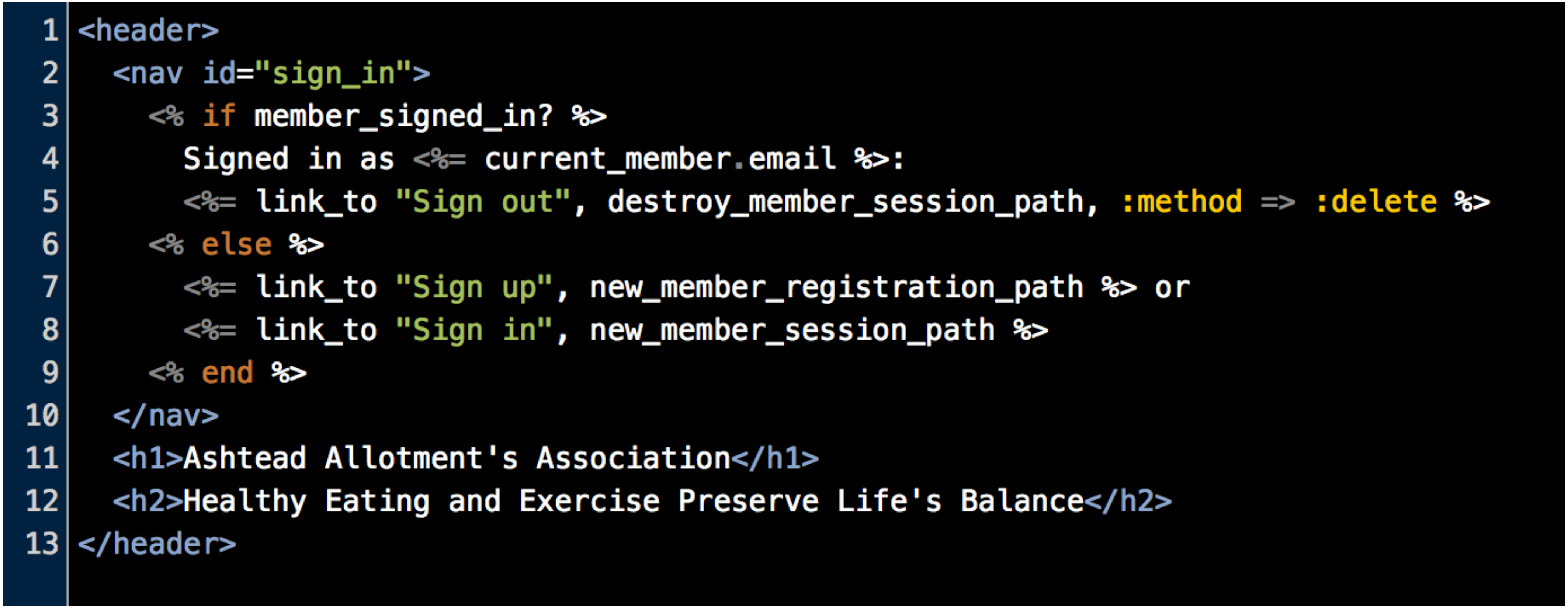
- *rails generate devise:views*
    - You will need to customise these
- *rails generate devise Member*
    - You can use any model name of your choice, not just *Member*
        - *User* and *Admin* for example
- And Don’t Forget!
    - *rake db:migrate*

rake routes

```
new_member_session            GET       /members/sign_in
destroy_member_session        DELETE    /members/sign_out
member_password               POST      /members/password
new_member_password           GET       /members/password/new
cancel_member_registration    GET       /members/cancel
new_member_registration       GET       /members/sign_up
```

N.B. Ensure that “member” corresponds
to the name of your Devise model

Signing out



```
<header>
  <nav id="sign_in">
    <% if member_signed_in? %>
      Signed in as <%= current_member.email %>:
      <%= link_to "Sign out", destroy_member_session_path, :method => :delete %>
    <% else %>
      <%= link_to "Sign up", new_member_registration_path %> or
      <%= link_to "Sign in", new_member_session_path %>
    <% end %>
  </nav>
  <h1>Ashtead Allotment's Association</h1>
  <h2>Healthy Eating and Exercise Preserve Life's Balance</h2>
</header>
```

app/views/layouts/_header.html.erb

What we have done

- We have:
  - added validations to our input forms
  - added an authorisation layer to our application