Lesson 10:

Creating forms for models

Objectives

Generate scaffolding for a model
Understand REST and Rails scaffolding
Understand and create form field elements with helpers
Create custom helper methods

Recap

How do you access form field parameters within a controller?

Using the `params` hash

Recap

What are some form helper methods?

`form_tag`, `text_field_tag`, `submit_tag`

Recap

What is a Rails model?

A class derived from ActiveRecord::Base representing one record from a table and enabling access to that table

Recap

What is a Rails migration?

A class derived from ActiveRecord::Migration defining how a table should be created

Recap

What is the purpose of Rake?

To execute a set of tasks defined in a rakefile

Recap

How might you retrieve all reviews for a movie named 'Tron'?

Movie.where(movie_name: "Tron")

Generating scaffolding for a model

What is Rails scaffolding?

Rails supports generating a scaffold for a particular model

```
rails generate scaffold <model_name>
<column_name:data_type>…
```

A movie review application could be quickly generated using:

```
rails generate scaffold Review title:string rating:float
user_name:string
```

What is Rails scaffolding?

Generating a scaffold for a model creates:

– A model class exposing the relevant accessors
– A migration to create the relevant tables
– A route configuration to map requests to the controller
– A controller to move data between the components
– Four views to show a record, edit it, list all records, or add a new record
– Additional supporting files for scripting, CSS, testing, and helper methods

How is scaffolding using?

Scaffolding is useful for:
– simple applications and client demonstrations
– demonstrating best practices regarding REST based design (more on this ahead)

But scaffolding may be too simplistic for some applications

Exercise:

Creating a Rails app using scaffolding

Understanding REST & Rails scaffolding

What is REST?

REST stands for REpresentational State Transfer

What is REST?

Web applications traditionally use only the HTTP GET and POST methods from the browser:

`GET`
data displayed in the URL query string

`POST`
data hidden in the request packet

What is REST?

REST is an approach to web app development making full use of five standard HTTP protocol methods

GET, POST, PUT, DELETE and HEAD

Using only GET and POST can lead to ugly URLs, unbookmarkable applications, caching issues, and related problems.

What is REST?

RESTful URLs should always cause the same result and provide the same output

Rails supports older browsers which may not support PUT and DELETE using hidden form field based workaround

How does Rails handle RESTful requests?

Scaffolding adds a resources entry to config/routes.rb:

```
resources :reviews
```

A resources entry equates to a set of routes defined for the Reviews resource

All routes defined in an application can be viewed using the `rake routes` command

What actions does a RESTful controller provide?

Rails provides 7 actions when generating a REST-ful Application Programming Interface.

What actions does a RESTful controller provide?

index	return *all* model data, list it in the requested format
show	display one model object specified by id
new	create a new empty model object, display in a form
create	create new model object using values passed as parameters, display success or failure
edit	return record specified by id, display it an editable form
update	retrieve record specified by id, update it using values passed as parameters, display success or failure
destroy	delete record specified by id then redirect to index

What is the :format value in a Rails URL?

What appear as file extensions (e.g., .html, .json) in a URL are assigned to a format variable within the controller.

The respond_to method of a controller can examine the format variable and determine how to format the response

```
def show
    @review = Review.find(params[:id])

    respond_to do |format|
        format.html # shows show.html.erb view
        format.json { render json: @book }
    end
end
```

Exercise:
Understanding RESTful URLs

Understanding & creating form field elements

What views does Rails scaffolding create by default?

Rails scaffolding generates four views for a model

Rails scaffolding also generates one partial view

What is a partial view?

A partial enables the same code to be reused in several places

Remeber to stay DRY (Don't Repeat Yourself)

Partials are:

saved with a preceding underscore in their file name
e.g. _form.html.erb

displayed using the render command in an action or view without the preceding underscore:

```
<%= render form %>
```

Exercise:

Create and review multi-field forms

What is the form_for command?

A FormHelper (*a "helper method"*) is designed to help build and render HTML content

`form_for` is a helper which understands models and know how to generate HTML forms for them:

– if a model is empty, a blank input form is generated
– if a model has value, the values are displayed in the form fields for editing

What is the form_for command?

For most controls, Rails helper methods generate the code needed to set the field value when editing the form:

Rails does not generate radio buttons, though, so this behavior must be built manually

What is the difference between model and non-model form helpers?

Non-model based form helpers:

```
<%= form_tag("/price_lookup", :method => "get") do %>
    <%= label_tag(:product_id, "Product ID:") %>
    <%= text_field_tag(:product_id) %>
    <%= submit_tag("Price Lookup") %>
<% end %>
```

Model based form helpers:

```
<%= form_for(@product) do |f| %>
    <%= f.label(:product_id) %>
    <%= f.text_field(:product_id) %>
    <%= f.submit %>
<% end %>
```

What helper methods are built in?

Both model ("form_tag") and non-model ("form_for" and "f." prefix) helpers are available for each of these:

label	date_select
text_field	time_select
text_area	datetime_select
check_box	password_field
radio_button	submit
select	hidden

How does a label helper work?

Helper method for a model named Review (f is a reference to the form_for object itself)

```
<%= f.label :title %>
<%= f.text_field :title %>
```

Generated HTML, label identified as for the corresponding review_title input tags:

```
<label for="review_title">Title</label>
<input id="review_title" name="review[title]" size="30"
type="text" />
```

How does the text_field helper work?

Helper method for a model named Review

```
<%= f.text_field :title %>
```

Generated HTML (field named review[title] to identify the model variable assigned to it)

```
<input id="review_title" name="review[title]" size="30"
type="text" />
```

How does the text_area helper work?

Helper method for a model named Review, could optionally specify `cols: value` and `rows: value`

```
<%= f.text_area :synopsis, rows: 8 %>
```

Generated HTML:

```
<textarea cols="40" id="review_synopsis"
name="review[synopsis]" rows="8"></textarea>
```

How does a checkbox helper work?

Helper method for a model named Review. As with all HTML helpers, could optionally assign CSS class as `css: 'class_name'`

If needed, can override checked status as `checked: 'checked'`

```
<%= f.check_box :status %>
```

Generated HTML, corresponding hidden field ensures value sent even if unchecked:

```
<input name="review[status]" type="hidden" value="0" />
<input id="review_status" name="review[status]"
type="checkbox" value="1" />
```

How does a date_select helper works?

date_select, time_select, and datetime_select each work similarly to this example

```
<%= f.date_select :release %>
```

```
<select id="review_release_1i" name="review[release(1i)]">
    <option value="2007">2007</option>
    ... (current marked selected)
    <option value="2017">2017</option>
</select>
<select id="review_release_2i" name="review[release(2i)]">
    <option value="1">January</option>
    ... (current marked selected)
    <option value="12">December</option>
</select>
<select id="review_release_3i" name="review[release(3i)]">
    <option value="1">1</option>
    ... (current marked selected)
    <option value="31">31</option>
</select>
```

How does a number helper work?

Helper method for a model named Review

```
<%= f.number_field :rating %>
```

Generated HTML, server side validation errors distinguishing float, decimal, and integer display at top of form

```
<input id="review_rating" name="review[rating]"
type="number" />
```

How could you create a radio button set?

There is a radio_button helper method, but Rails does not generate radio button sets

How could you create a radio button set?

Set the radio button labels and values as a hash in the controller

```
def new
    @ratings = {
      "G" => "General Admission",
      "PG" => "Parental Guidance",
      "R" => "Restricted",
      "X" => "Adults Only"
    }
end
```

How could you create a radio button set?

Loop over the hash to generate the radio buttons and use a condition to set which is checked in the form:

```
<fieldset>
    <legend>Rating</legend>

    <% @ratings.each do |label, code| %>
        <% if :rating == code %>
            <%= f.radio_button :rating, label, :checked %>
        <% else %>
            <%= f.radio_button :rating, label %>
        <% end %>

        <label for="<%= 'movie_rating_' + code %>">
            <%= label %>
        </label>
    <% end %>
</fieldset>
```

How could you manually populate a select control?

Like radio buttons, there is a select helper method, but Rails will automatically not generate it

How could you manually populate a select control?

Set the select labels and values as a hash in the controller

```
def new
    @ratings = {
      "G" => "General Admission",
      "PG" => "Parental Guidance",
      "R" => "Restricted",
      "X" => "Adults Only"
    }
end
```

How could you manually populate a select control?

Assign the has to the select helper

```
<%= f.select :rating, @ratings %>
```

How do you assign CSS or other attributes to a helper?

CSS or other inline attributes can be added to a Rails helper method

Attributes added to the helper will be added to the generated tag, or override attributes already to be set:

```
<%= f.text_field :name, class: “big bold red”,
size: “45” %>

<input id=”person_name” name=”person[name]” type=”text”
class=”big bold red” size=”45” />
```

How are input validation errors displayed?

Validation means to verify user input matches relevant data type and format rules.

e.g. is that thing you typed in an email address or not?

Validation is discussed in more detail later in the course

How are input validation errors displayed?

Rails scaffolding validates input on the server – not browser – and displays errors in _form.html.erb

```
<% if @review.errors.any? %>
    <div id="error_explanation">
        <h2>
            <%= pluralize(@review.errors.count, "error") %>
            prohibited this review from being saved:
        </h2>
        <ul>
            <% @review.errors.full_messages.each do |msg| %>
            <li><%= msg %></li>
            <% end %>
        </ul>
    </div>
<% end %>
```

Exercise:
Customizing a Rails scaffolding form

Creating helper methods

What is a custom form helper method?

A rails form helper method generates HTML content and returns it to the view where it is called

Rails generates helper method modules for the application and models in the app folder

– application_helper.rb: methods available in all views
– [model_name]_helper.rb: methods available in views presented for this model and its controller

How does the radio_button helper work?

So far, each helper method discussed has been called within a form_for block:

```
<%= form_for(@review) do |f| %>
    ...
    <%= f.radio_button :rating, label, :checked %>
    ...
<% end %>
```

How does the radio_button helper work?

Each of the helper methods can be also called independently of the form_for helper (e.g., in a custom helper)

```
radio_button(model_name, target_property, value)
```

The radio_button helper marks a radio button as checked if target_property matches the value

How could you write a custom helper?

A custom helper takes form values, generates an appropriate string of HTML, and returns the HTML

Built in form helpers can be called by a custom form helper.

Custom helper method defined in app/helpers/reviews_helper.rb

How could you write a custom helper?

```
module ReviewsHelper

    # new action we've made
    def radio_buttons(model_name, source_hash, target_property, legend)
        # html variable holds a string to output
        html = ''
        html << '<fieldset><legend>' + legend + '</legend>'

        # source_hash is the PG, X ratings, etc
        source_hash.each do |key, value|
            html << radio_button(model_name, target_property, value)
            html << label(target_property, value)
        end

        html << '</fieldset>'

        # return the html string
        return html.html_safe
    end
end
```

How could you write a custom helper?

```
<%= radio_buttons("review", @ratings, :rating, "Rating") %>
```

What was that html_safe method?

Rendering string values into a browser *can* be dangerous

Rails strings support a boolean (*true or false*) .html_safe assertion, enabling strings to be flagged as being safe for display

If a string is not flagged as safe, Rails views may escape angle brackets, etc. to prevent malicious script from running.