

In this lecture, we will discuss...

- ✧ Partials
- ✧ How the “form” partial works

Partials: DRY (Don't Repeat Yourself)

- ✧ Rails encourages the DRY principle
- ✧ We already know about the `application.html.erb`, which enables you to **maintain layout code** for the entire application in **one place** (more on this later)
- ✧ It would also be nice to **reuse** snippets of view code in **multiple templates**
- ✧ For example, `edit` and `new` forms – are they really that much different?



Partials

- ✧ Partials are **similar** to regular templates, but they have a more **refined** set of capabilities
- ✧ Named with **underscore** (`_`) as the leading character
- ✧ Rendered with **render** `'partialname'` (no underscore)
- ✧ **render** also accepts a **second argument**, a hash of **local** variables used in the partial



Object Partial

- ✧ Similar to passing local variables, you can also **render** a specific object
- ✧ `<%= render @post %>` will render a partial in `app/views/posts/_post.html.erb` and automatically assign a local variable `post`



Convention Over Configuration

Rendering Collection of Partial

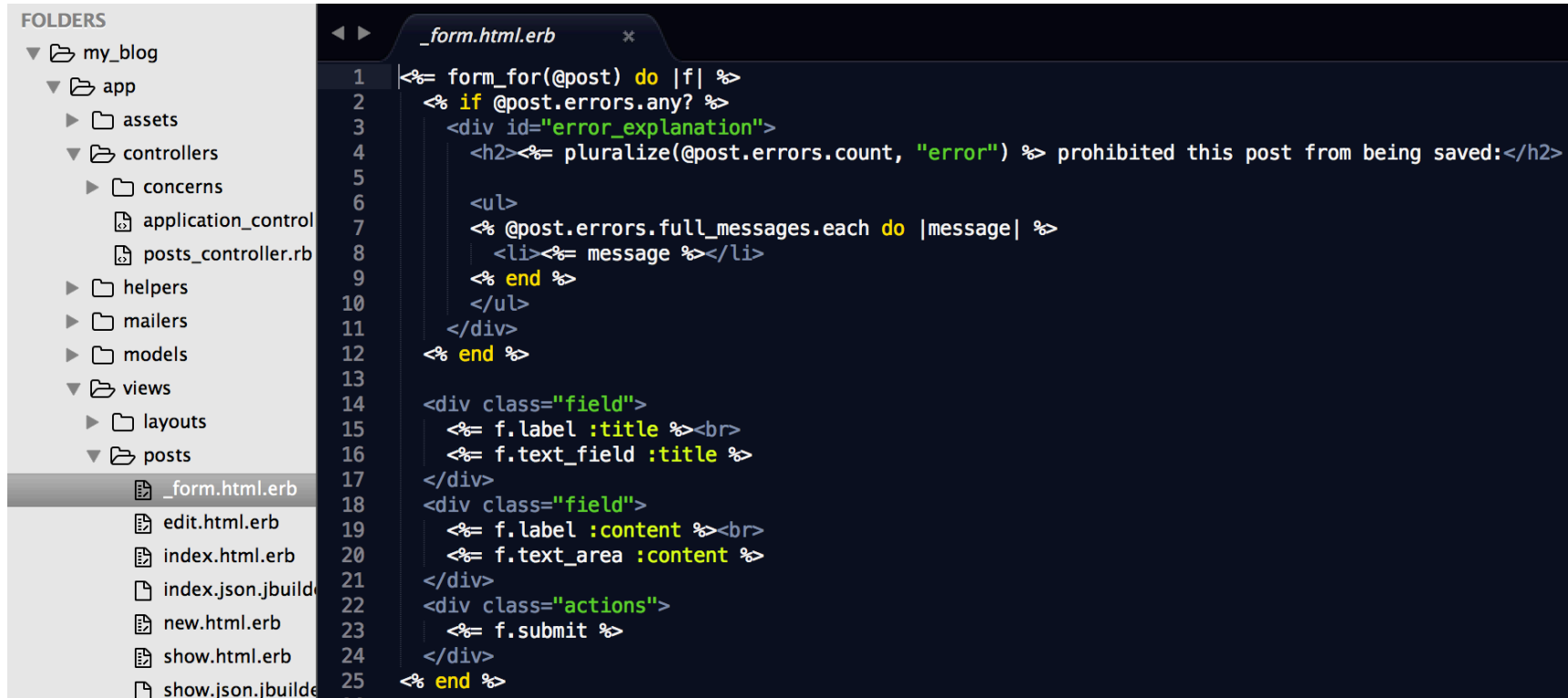
```
<%= render @posts %>
```

is equivalent to

```
<% @posts.each do |post| %>  
  <%= render post %>  
<% end %>
```



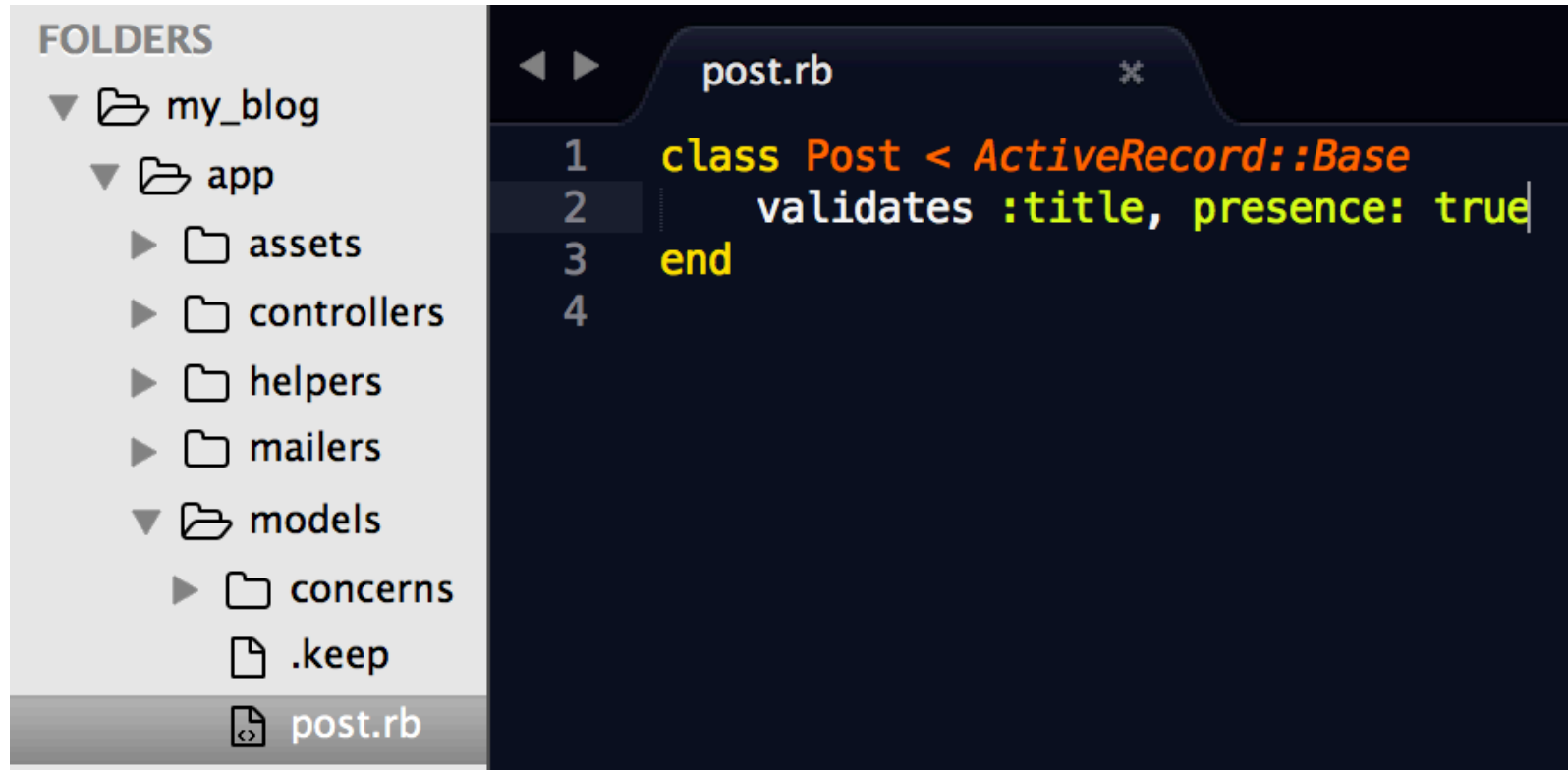
_form.html.erb – display errors



```
1 <%= form_for(@post) do |f| %>
2   <% if @post.errors.any? %>
3     <div id="error_explanation">
4       <h2><%= pluralize(@post.errors.count, "error") %> prohibited this post from being saved:</h2>
5
6       <ul>
7         <% @post.errors.full_messages.each do |message| %>
8           <li><%= message %></li>
9         <% end %>
10      </ul>
11    </div>
12  <% end %>
13
14  <div class="field">
15    <%= f.label :title %><br>
16    <%= f.text_field :title %>
17  </div>
18  <div class="field">
19    <%= f.label :content %><br>
20    <%= f.text_area :content %>
21  </div>
22  <div class="actions">
23    <%= f.submit %>
24  </div>
25 <% end %>
```



Require Presence of Title



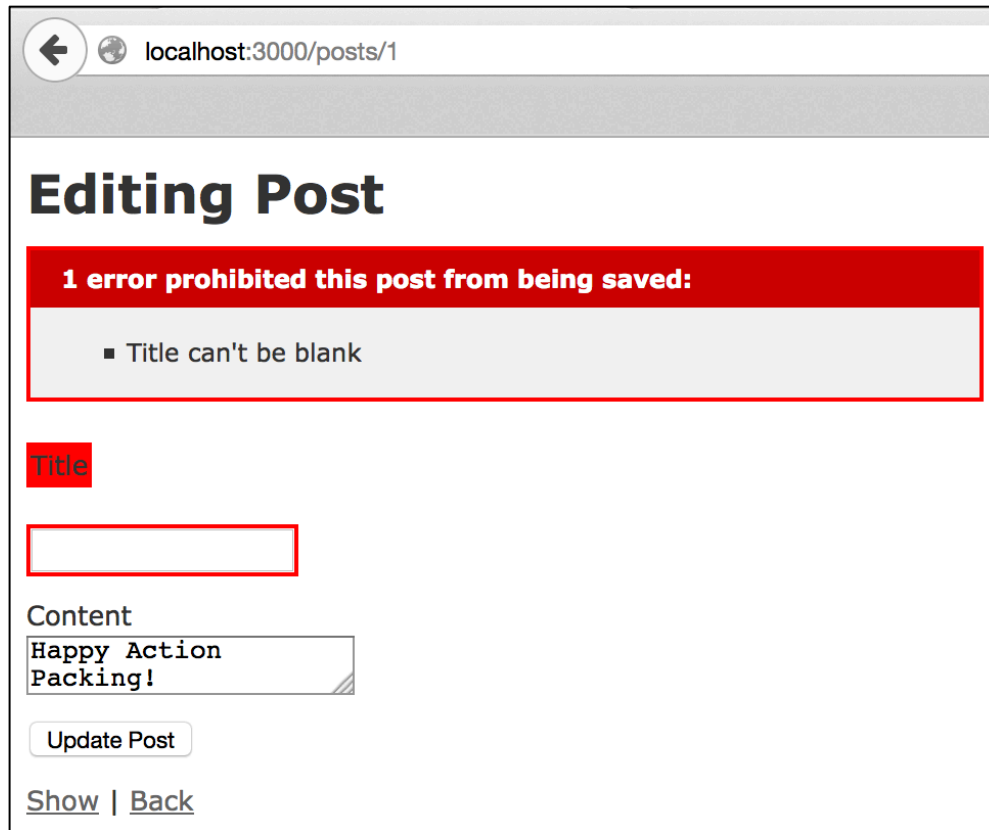
The image shows a code editor interface. On the left is a sidebar titled "FOLDERS" showing a file tree. The tree structure is as follows:

- my_blog
 - app
 - assets
 - controllers
 - helpers
 - mailers
 - models
 - concerns
 - .keep
 - post.rb

The main editor area shows the content of "post.rb" with the following code:

```
1 class Post < ActiveRecord::Base
2   validates :title, presence: true
3 end
4
```

_form.html.erb – display errors



A screenshot of a web browser window showing a form for editing a post. The browser's address bar displays 'localhost:3000/posts/1'. The page title is 'Editing Post'. A prominent red error message box states '1 error prohibited this post from being saved:'. Below this, a list of errors shows 'Title can't be blank'. The 'Title' label is highlighted in red, and the corresponding text input field is outlined with a red border. The 'Content' label is followed by a text area containing the text 'Happy Action Packing!'. At the bottom of the form is an 'Update Post' button. Below the form, there are links for 'Show' and 'Back'.

localhost:3000/posts/1

Editing Post

1 error prohibited this post from being saved:

- Title can't be blank

Title

Content

Happy Action Packing!

Update Post

[Show](#) | [Back](#)

Summary

- ✧ Partial is a snippet of reusable template that has an underscore in its name and accepts parameters when rendered

What's Next?

- ✧ Form helpers and Layouts

