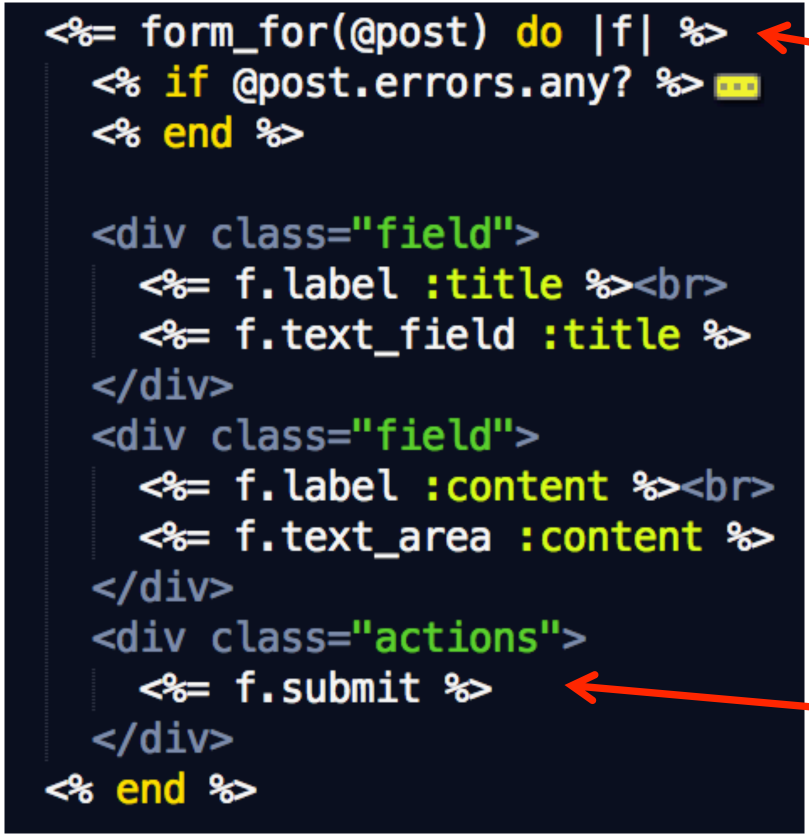
In this lecture, we will discuss…

- Form helpers
- Layouts

_form.html.erb



```
<%= form_for(@post) do |f| %>
  <% if @post.errors.any? %>
  <% end %>

  <div class="field">
    <%= f.label :title %><br>
    <%= f.text_field :title %>
  </div>
  <div class="field">
    <%= f.label :content %><br>
    <%= f.text_area :content %>
  </div>
  <div class="actions">
    <%= f.submit %>
  </div>
<% end %>
```

Form with parameters that match up with model's attributes

Submit button for submitting the form

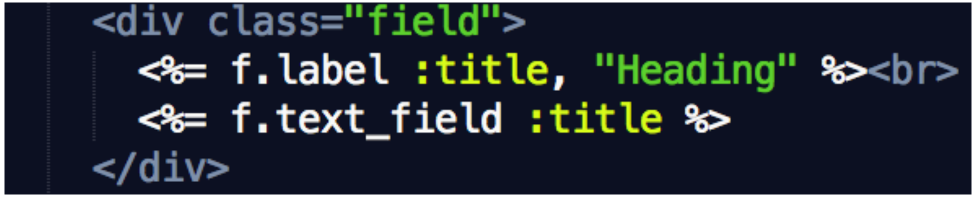
Form Helpers

- `form_for`
  - Generates a form tag for passed in object
  - Unlike a regular HTML form, Rails uses POST by default
  - This of course makes a lot of sense:
    1. Your password is not passed as part of your URL
    2. Anything that will end up modifying data on the server should definitely be a POST and not GET

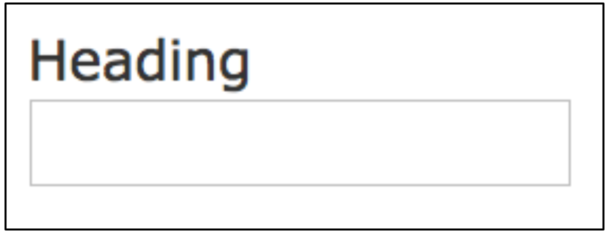
Form helpers – f.label

✧ `f.label`

- Outputs HTML label tag for the provided attribute
- To customize label description, pass in a string as a second parameter



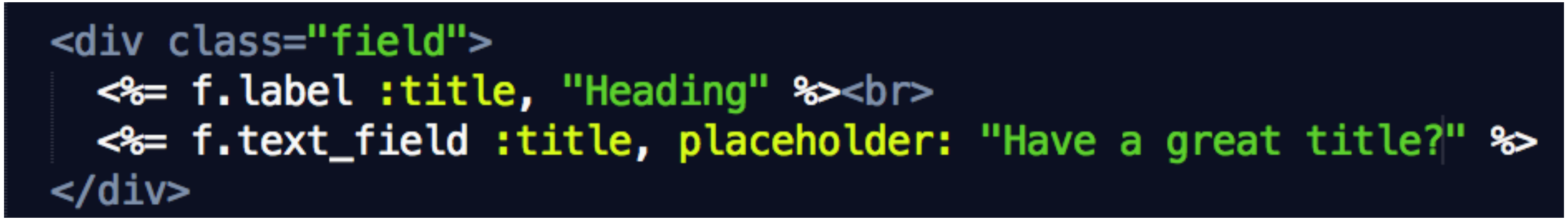
```
<div class="field">
  <%= f.label :title, "Heading" %><br>
  <%= f.text_field :title %>
</div>
```



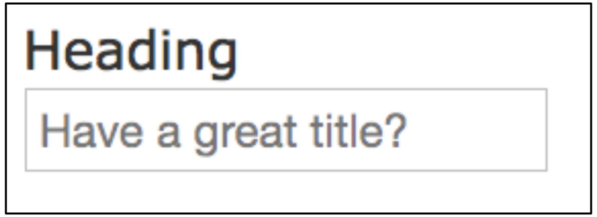
Form Helpers – f.text_field

✧ `f.text_field`

- Generates input type="text" field
- Use `:placeholder` hash entry to specify a placeholder (hint) to be displayed inside the field until the user provides a value



```
<div class="field">
  <%= f.label :title, "Heading" %><br>
  <%= f.text_field :title, placeholder: "Have a great title?" %>
</div>
```



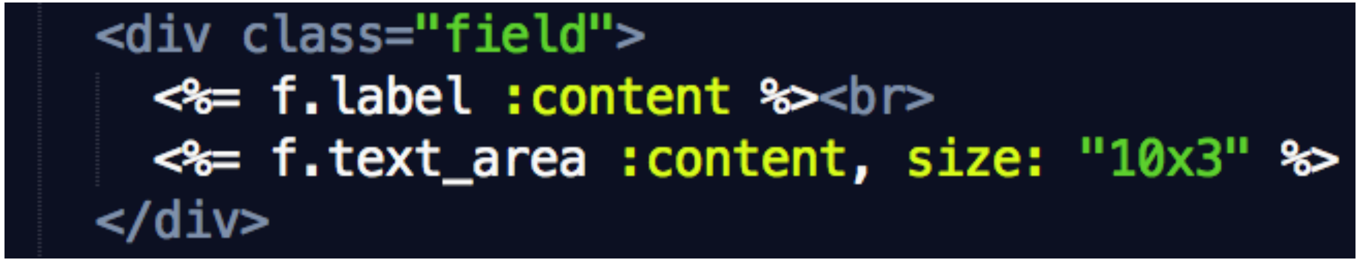
Heading

Have a great title?

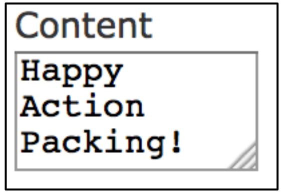
Form Helpers – f.text_area

✧ `f.text_area`

- Similar to `f.text_field`, but for a text area instead of a text field input (default: 40 cols x 20 rows)
- Can specify a different size (colsXrows) with a :size attribute



```
<div class="field">
  <%= f.label :content %><br>
  <%= f.text_area :content, size: "10x3" %>
</div>
```



Content

```
Happy
Action
Packing!
```

Date Helpers

- `f.date_select`
  - Set of select tags (year, month, day) pre-selected for accessing an attribute in the DB. Many formatting options `f.time_select`
- `f.datetime_select`
- `distance_of_time_in_words_to_now`
- And many many more…
- See ActionView::Helpers::DateHelper docs
  - http://api.rubyonrails.org/classes/ActionView/Helpers/DateHelper.html

Form Helpers – Others

- `search_field`
- `telephone_field`
- `url_field`
- `email_field`
- `number_field`
- `range_field`

Some of these are browser-dependent – will take advantage of the browsers that are ready for prime time and will still look okay in others…

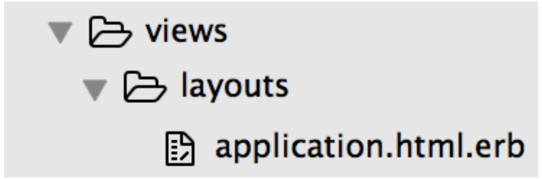
Form Helpers – f.submit

- `f.submit`
  - Submit button
  - Accepts the name of the submit button as its first argument
  - If you don't provide a name – generates one based on the model and type of action, e.g. "Create Post" or "Update Post"

http://guides.rubyonrails.org/form_helpers.html

More on Layouts

1. Layout named `application.html.erb` is applied by default as a shell for any view template



2. Layout that matches the name of a controller is applied if present (overriding 1. above)
3. You can use `layout` method inside controller (outside any action) to set a layout for the entire controller

```
layout 'some_layout'
```

Layouts During Rendering

- ✧ You can include a layout for a specific action with an explicit call to `render` inside the action
  `render layout: 'my_layout'`
- ✧ If you don't want a layout (for some reason) – just pass false instead of layout name `render layout: false`

Summary

- Form helpers are a quick way to generate forms as well as form elements
- Layouts let you display a common “shell” around application template or around particular actions or resources