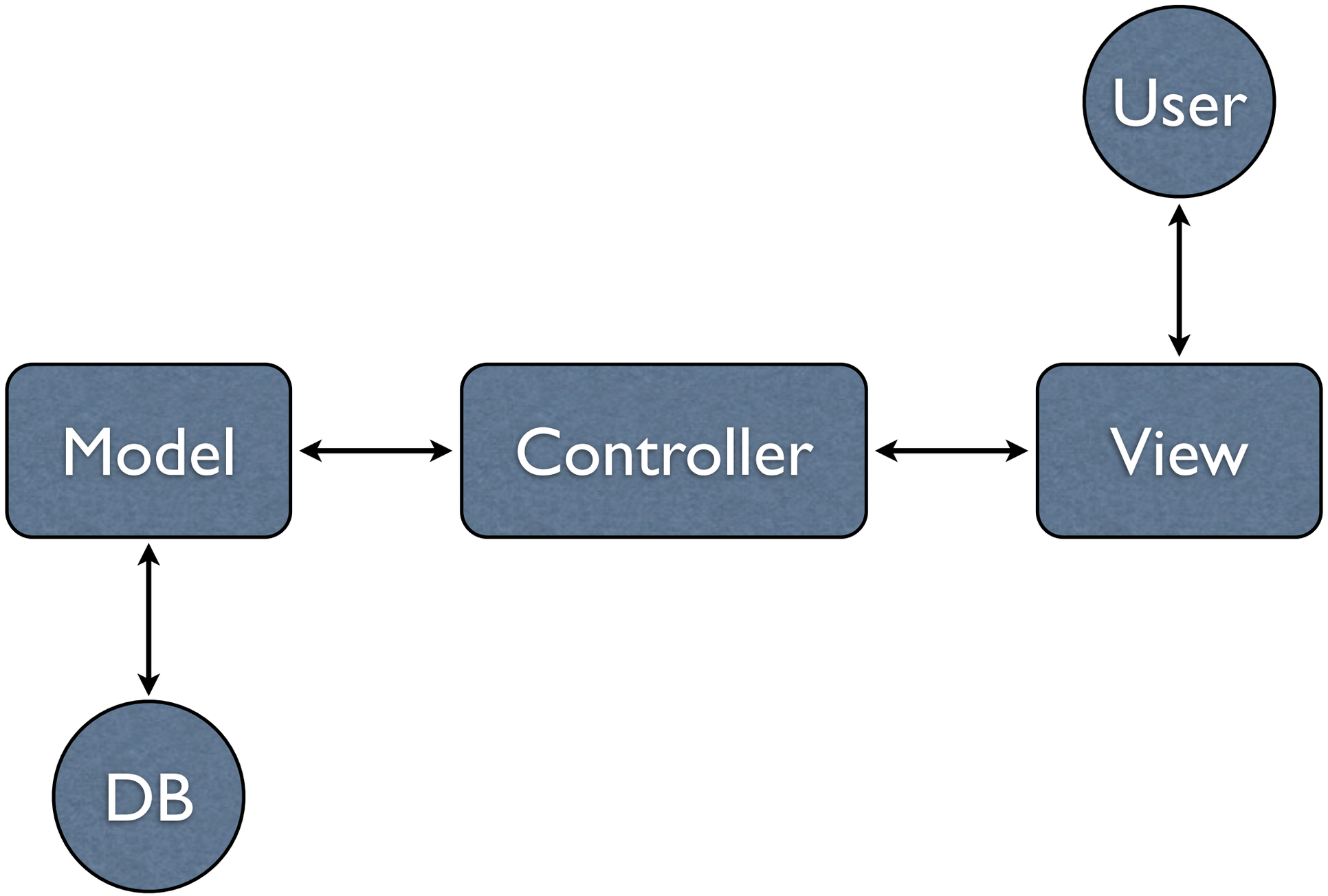
Beginning Ruby on Rails

Session Four


User
Model
Controller
View
DB

All About Views

- ERB
- Helpers
- Layouts
- Partials
- Forms

ERB

Embedded Ruby

ERB Templates

- ERB is the default type of view template
  - Another popular choice is Haml
- ERB lets you mix Ruby code with HTML
  - Similar to ASP, JSP, or PHP

Output

- Edit app/views/posts/show.html.erb
- Output text, convert HTML:

```
<%= @post.title %>
```

- Output raw text, can be dangerous:

```
<%= raw @post.body %>
```

Code

- Edit app/views/posts/index.html.erb
- Looping over an array of posts:

```
<% @posts.each do |post| %>

    ...

<% end %>
```

Comments

- Notes to yourself or other programmers
- Will not appear in HTML source:

```
<%# This code is crazy %>
```

Helpers

Simplify views and remove duplication

URL Helpers

- Create links with link_to:

```
link_to ‘Show’, post

link_to ‘Edit’, edit_post_path(post)

link_to ‘Destroy’, post, :confirm =>
‘Are you sure?’, :method => :delete
```

Number Helpers

- Handy methods for displaying numbers:

  `number_to_currency`

  `number_to_human`

  `number_to_percentage`

  `number_with_precision`

Your Own Helpers

- Edit app/helpers/application_helper.rb

```
module Application Helper
  def friendly_date(d)
    d.strftime("%B %e, %Y")
  end
end
```

Layouts

Base HTML for the web site

Application Layout

- Edit app/view/layouts/application.html.erb
- Contains the basic HTML for a page
- Links to stylesheets
- Includes Javascript

Asset Tags

- Assets are in the public directory
  - Error pages, images, css, javascript

`stylesheet_link_tag`

`javascript_include_tag`

yield

- In a layout, the yield statement identifies where content from the view should be inserted.
- It is possible to have more than one yield per layout:
  - For example, `<%= yield :head %>` is often used in the page header.

content_for

- The content_for statement denotes content for a named yield block:

```
<% content_for(:head) do %>
    . . .
<% end %>
```

Partials

Share code between pages

Partials

- Repeated parts of pages are commonly separated out into partials.
- Partials are view templates that begin with an underscore.
- For example, _form.html.erb

```
<%= render ‘form’ %>
```

Collections

- Can eliminate loops in view templates.
- The partial is inserted for each member.

```
<%= render :partial => ‘post’,
  :collection => @posts %>
or simply:
<%= render @posts %>
```

Showing Comments

- Let's add the comments to the show page.

```
<% @post.comments.each do |comment| %>
  <p><%= comment.author %> said:</p>
  <blockquote><p><%= comment.body %></p></blockquote>
<% end %>
```

Forms

Get input from users

form_for

- The form_for block is used to create a form bound to a model.
- Within this block, you have access to methods to build controls on the form.
- For example, `f.text_field :title` tells Rails to create a text input, and connect it to the title attribute.

Form Helpers

- label
- text_field
- text_area
- submit

form_tag

- The form_tag block is used to create a form not bound to a model.
- All form helpers used with form_tag include _tag at the end
  - For example, label_tag and text_field_tag

Comment Form

```
<h3>Add A Comment</h3>
<%= form_for([@post, @post.comments.build]) do |f| %>
  <div class="field">
    <%= f.label :author %><br />
    <%= f.text_field :author %>
  </div>
  <div class="field">
    <%= f.label :body %><br />
    <%= f.text_area :body %>
  </div>
  <div class="actions">
    <%= f.submit %>
  </div>
<% end %>
```

Homework

There may be a test next time...

Homework

- We're going to build a new app for the second half of the class. I am thinking building a Twitter clone. If you'd rather we build something else, let me know.
- Make sure you understand everything inside the "app" directory and how models, views and controllers interact.