First Project Meeting and Basic Forms (Addressbook)

Siddharth Kaza
Towson University, Computer and Information Sciences

Outline

- Mostly coding today!
- Basic forms
  - AddressBook Mini project
- First project meeting
  - Making stories and dividing work

Forms

- Forms in web applications are an essential interface for user input.
- But, markup can quickly become tedious to write and maintain.
- Frameworks deal away with these complexities by providing view helpers for generating form markup.
- Use cases
  - Forms without models (e.g., search)
  - Forms with models (e.g., product entry in store)

Forms without scaffolding

- We will be learning to use forms without the scaffolding first.

- Then scaffolding as shown in the book.

Basic form helper

The most basic form helper is `form_tag`.

```
<%= form_tag do %>
  Form contents
<% end %>
```

When called without arguments like this, it creates a `<form>` tag which, when submitted, will POST to the current page. For instance, assuming the current page is `/home/index`, the generated HTML will look like this (some line breaks added for readability):

```
<form accept-charset="UTF-8" action="/home/index" method="post">
  <div style="margin:0;padding:0">
    <input name="utf8" type="hidden" value="&#x2713;" />
    <input name="authenticity_token" type="hidden" value="f755bb0ed134b
  </div>
  Form contents
</form>
```

Forms without models

```
<%= form_tag("/search", :method => "get") do %>
  <%= label_tag(:q, "Search for:") %>
  <%= text_field_tag(:q) %>
  <%= submit_tag("Search") %>
<% end %>
```

URL

Method

This will generate the following HTML:

```
<form accept-charset="UTF-8" action="/search" method="get">
  <label for="q">Search for:</label>
  <input id="q" name="q" type="text" />
  <input name="commit" type="submit" value="Search" />
</form>
```

Label allows mouse users to click on it select the input area

Binding a form to a model

- form_for binds a form to a model

Assume we have a controller for dealing with articles `app/controllers/articles_controller.rb`:

```
def new
  @article = Article.new
end
```

The corresponding view `app/views/articles/new.html.erb` using `form_for` looks like this:

```
<%= form_for @article, :url => { :action => "create" }, :html => {:clas
  <%= f.text_field :title %>
  <%= f.text_area :body, :size => "60x12" %>
  <%= f.submit "Create" %>
<% end %>
```

The resulting HTML is:

```
<form accept-charset="UTF-8" action="/articles/create" method="post" cl
  <input id="article_title" name="article[title]" size="30" type="text"
  <textarea id="article_body" name="article[body]" cols="60" rows="12">
  <input name="commit" type="submit" value="Create" />
</form>
```

Within the depot context (chapter 12)

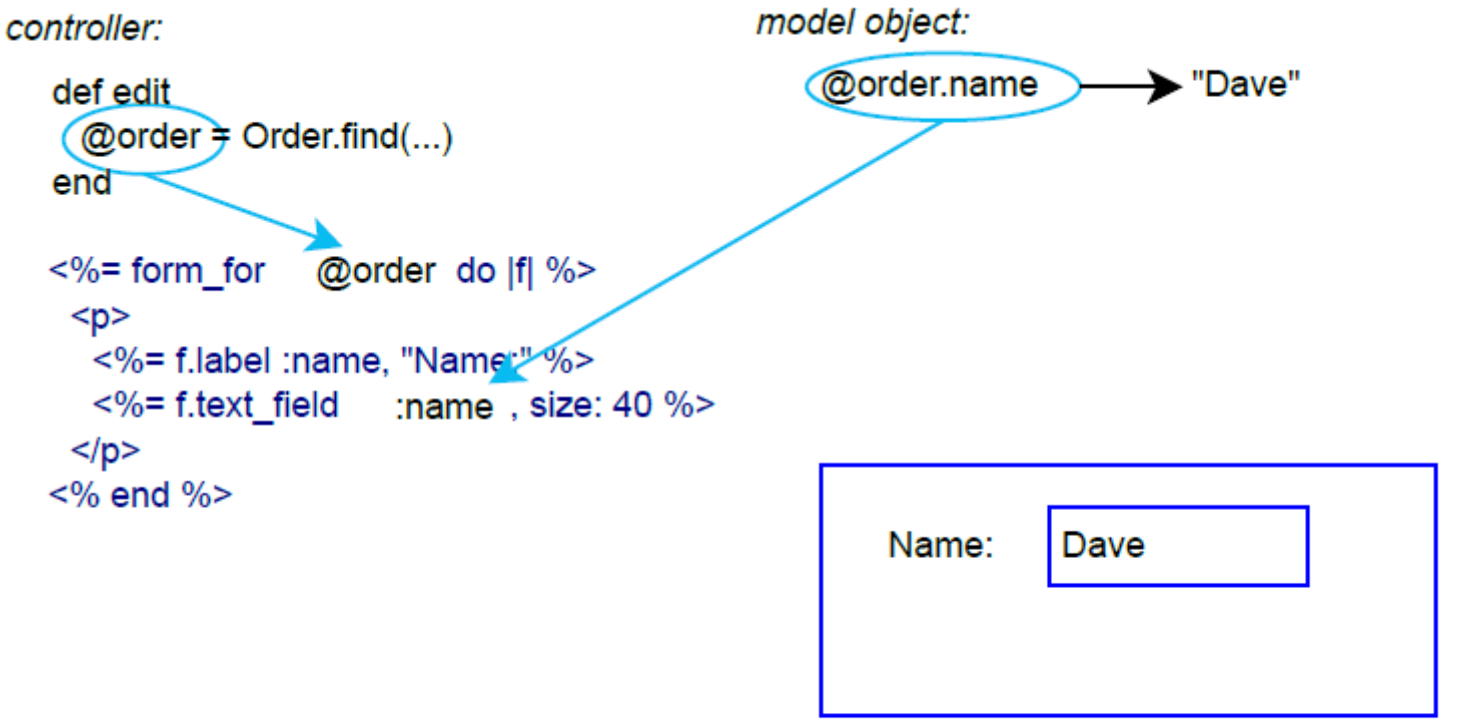

Figure 21—Names in form_for map to objects and attributes

Web Address Book - Design

- Look at the web addressbook stories
  - Focus on
    - what controllers you need
    - what actions you need within those controllers
    - what models you need
- Ask me questions – this should prepare you for your exam.

Web Addressbook demo

- Show the code
  - Go through the story and show the commands
  - Show controllers
  - Remember adding all the right routes

Using Scaffolding

- A **scaffold** in Rails is a full set of model, database migration for that model, controller to manipulate it, views to view and manipulate the data, and a test suite for each of the above. (http://guides.rubyonrails.org/command_line.html)

Twitter Bootstrap

- What is it: Sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development.
  - http://getbootstrap.com/
  - Provides CSS and Javascript

- You can use a gem to include it into rails (or include it in its native form)
  - We will use Twitter-bootstrap-rails https://github.com/seyhunak/twitter-bootstrap-rails
  - Also see http://railscasts.com/episodes/328-twitter-bootstrap-basics

Twitter Bootstrap Demo

```
gem "twitter-bootstrap-rails"
```

After running `bundle install`, run the generator:

```
rails generate bootstrap:install static
```

Example of a responsive layout:

```
rails g bootstrap:layout application fluid
```

Themed (generates Twitter Bootstrap compatible scaffold vi

Usage:

```
rails g bootstrap:themed [RESOURCE_NAME]
```

Example:

```
rails g scaffold Post title:string description:text
rake db:migrate
rails g bootstrap:themed Posts
```

For this to work for the AdddressBook, you will have the change the

<%- model_class = Addres -%>
To
<%- model_class = Address -%>

in each view. This specifies the name of the model class.

Inflections

To use scaffolding for the some projects, (e.g. Addressbook) we may have to first override the inflector that transforms 'Address' to its plural.

used for creating models, tables, etc.

Address Model -> addresses table

- The inflector class in rails transforms singular to plural, model names to table names etc.

  (http://api.rubyonrails.org/classes/ActiveSupport/Inflector.html)

- It is primarily needed so we can follow the convention over configuration paradigm