```
  include ApplicationHelper
  .
  .
  .
end
```

Listing 5.36: Using the `full_title` helper in a test. GREEN
test/integration/site_layout_test.rb

```
require 'test_helper'

class SiteLayoutTest < ActionDispatch::IntegrationTest

  test "layout links" do
    get root_path
    assert_template 'static_pages/home'
    assert_select "a[href=?]", root_path, count: 2
    assert_select "a[href=?]", help_path
    assert_select "a[href=?]", about_path
    assert_select "a[href=?]", contact_path
    get contact_path
    assert_select "title", full_title("Contact")
  end
end
```

Listing 5.37: A direct test of the `full_title` helper.
test/helpers/application_helper_test.rb

```
require 'test_helper'

class ApplicationHelperTest < ActionView::TestCase
  test "full title helper" do
    assert_equal full_title,         FILL_IN
    assert_equal full_title("Help"), FILL_IN
  end
end
```

5.4 User signup: A first step

As a capstone to our work on the layout and routing, in this section we'll make a route for the signup page, which will mean creating a second controller along

the way. This is a first important step toward allowing users to register for our site; we'll take the next step, modeling users, in Chapter 6, and we'll finish the job in Chapter 7.

5.4.1 Users controller

We created our first controller, the Static Pages controller, in Section 3.2. It's time to create a second one, the Users controller. As before, we'll use **generate** to make the simplest controller that meets our present needs, namely, one with a stub signup page for new users. Following the conventional REST architecture favored by Rails, we'll call the action for new users **new**, which we can arrange to create automatically by passing **new** as an argument to **generate**. The result is shown in Listing 5.38.

Listing 5.38: Generating a Users controller (with a **new** action).

```
$ rails generate controller Users new
      create  app/controllers/users_controller.rb
       route  get 'users/new'
      invoke  erb
      create    app/views/users
      create    app/views/users/new.html.erb
      invoke  test_unit
      create    test/controllers/users_controller_test.rb
      invoke  helper
      create    app/helpers/users_helper.rb
      invoke    test_unit
      invoke  assets
      invoke    scss
      create      app/assets/stylesheets/users.scss
```

As required, Listing 5.38 creates a Users controller with a **new** action (Listing 5.39) and a stub user view (Listing 5.40). It also creates a minimal test for the new user page (Listing 5.41).

Listing 5.39: The initial Users controller, with a **new** action.
app/controllers/users_controller.rb

```
class UsersController < ApplicationController

  def new
  end
end
```

Listing 5.40: The initial `new` view for Users.
app/views/users/new.html.erb

```
<h1>Users#new</h1>
<p>Find me in app/views/users/new.html.erb</p>
```

Listing 5.41: The generated test for the new user page. GREEN
test/controllers/users_controller_test.rb

```
require 'test_helper'

class UsersControllerTest < ActionDispatch::IntegrationTest

  test "should get new" do
    get users_new_url
    assert_response :success
  end
end
```

At this point, the tests should be GREEN:

Listing 5.42: GREEN

```
$ rails test
```

Exercises

Solutions to the exercises are available to all Rails Tutorial purchasers here.

To see other people's answers and to record your own, subscribe to the Rails Tutorial course or to the Learn Enough All Access Bundle.

1. Per Table 5.1, change the route in Listing 5.41 to use **`signup_path`** instead of **`users_new_url`**.
2. The route in the previous exercise doesn't yet exist, so confirm that the tests are now RED. (This is intended to help us get comfortable with the RED/GREEN flow of Test Driven Development (TDD, Box 3.3); we'll get the tests back to GREEN in Section 5.4.2.)

5.4.2 Signup URL

With the code from Section 5.4.1, we already have a working page for new users at /users/new, but recall from Table 5.1 that we want the URL to be /signup instead. We'll follow the examples from Listing 5.27 and add a **`get '/signup'`** rule for the signup URL, as shown in Listing 5.43.

Listing 5.43: A route for the signup page. RED
config/routes.rb

```
Rails.application.routes.draw do
  root 'static_pages#home'
  get  '/help',    to: 'static_pages#help'
  get  '/about',   to: 'static_pages#about'
  get  '/contact', to: 'static_pages#contact'
  get  '/signup',  to: 'users#new'
end
```

With the routes in Listing 5.43, we also need to update the test generated in Listing 5.38 with the new signup route, as shown in Listing 5.44.

Listing 5.44: Updating the Users controller test to use the signup route. GREEN
test/controllers/users_controller_test.rb

```
require 'test_helper'

class UsersControllerTest < ActionDispatch::IntegrationTest

  test "should get new" do
    get signup_path
    assert_response :success
  end
end
```

Next, we'll use the newly defined named route to add the proper link to the button on the Home page. As with the other routes, **`get 'signup'`** automatically gives us the named route **`signup_path`**, which we put to use in Listing 5.45. Adding a test for the signup page is left as an exercise (Section 5.3.2.)

Listing 5.45: Linking the button to the signup page.
app/views/static_pages/home.html.erb

```
<div class="center jumbotron">
  <h1>Welcome to the Sample App</h1>

  <h2>
    This is the home page for the
    <a href="https://www.railstutorial.org/">Ruby on Rails Tutorial</a>
    sample application.
  </h2>

  <%= link_to "Sign up now!", signup_path, class: "btn btn-lg btn-primary" %>
</div>

<%= link_to image_tag("rails.svg", alt: "Rails logo", width: "200"),
                      "https://rubyonrails.org/" %>
```

Finally, we'll add a custom stub view for the signup page (Listing 5.46).

Listing 5.46: The initial (stub) signup page.
app/views/users/new.html.erb

```
<% provide(:title, 'Sign up') %>
<h1>Sign up</h1>
<p>This will be a signup page for new users.</p>
```

With that, we're done with the links and named routes, at least until we add a route for logging in (Chapter 8). The resulting new user page (at the URL /signup) appears in Figure 5.11.

Exercises

Solutions to the exercises are available to all Rails Tutorial purchasers here.

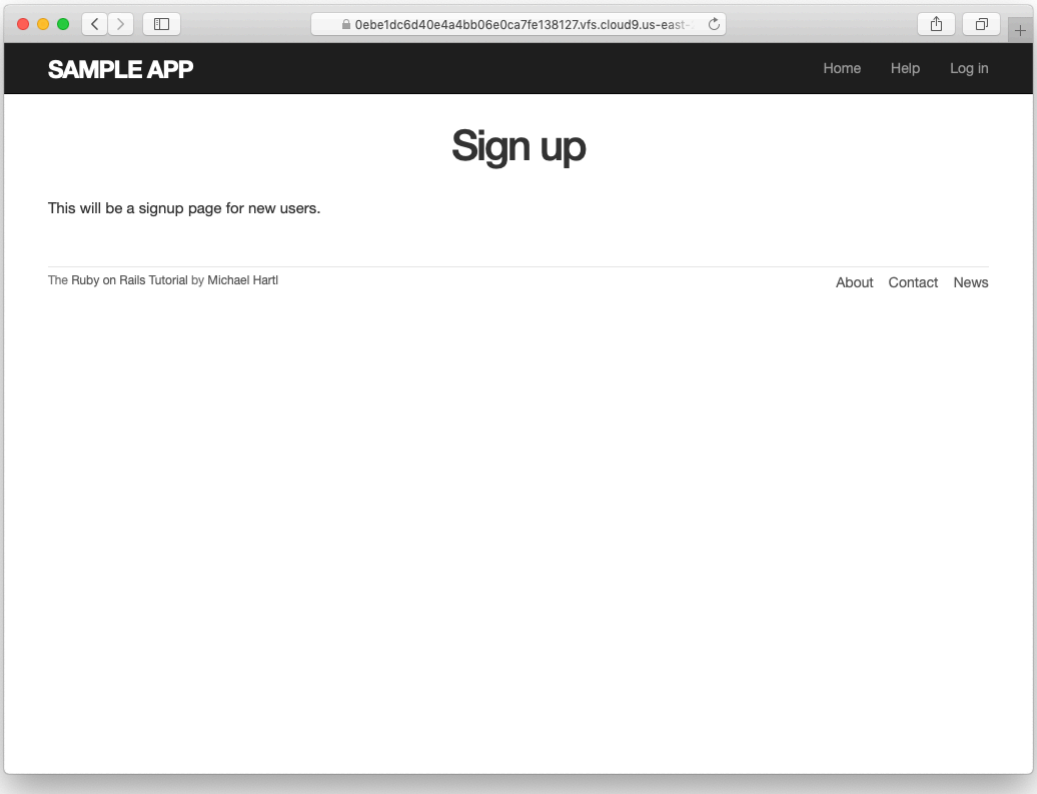

Figure 5.11: The new signup page at /signup.

To see other people's answers and to record your own, subscribe to the Rails Tutorial course or to the Learn Enough All Access Bundle.

1. If you didn't solve the exercise in Section 5.4.1, change the test in Listing 5.41 to use the named route **`signup_path`**. Because of the route defined in Listing 5.43, this test should initially be GREEN.

2. In order to verify the correctness of the test in the previous exercise, comment out the **`signup`** route to get to RED, then uncomment to get to GREEN.

3. In the integration test from Listing 5.32, add code to visit the signup page using the **`get`** method and verify that the resulting page title is correct. *Hint*: Use the **`full_title`** helper as in Listing 5.36.

5.5 Conclusion

In this chapter, we've hammered our application layout into shape and polished up the routes. The rest of the book is dedicated to fleshing out the sample application: first, by adding users who can sign up, log in, and log out; next, by adding user microposts; and, finally, by adding the ability to follow other users.

At this point, if you are using Git, you should merge your changes back into the master branch:

```
$ git add -A
$ git commit -m "Finish layout and routes"
$ git checkout master
$ git merge filling-in-layout
```

Then push up to GitHub (running the test suite first for safety):

```
$ rails test
$ git push
```

Finally, deploy to Heroku: