```
    get signup_path
    assert_difference 'User.count', 1 do
      post users_path, params: { user: { name:  "Example User",
                                         email: "user@example.com",
                                         password:              "password",
                                         password_confirmation: "password" } }
    end
    follow_redirect!
    assert_template 'users/show'
    assert is_logged_in?
  end
end
```

At this point, the test suite should still be GREEN:

Listing 8.32: 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. Is the test suite RED or GREEN if you comment out the **`log_in`** line in Listing 8.29?
2. By using your text editor's ability to comment out code, toggle back and forth between commenting out code in Listing 8.29 and confirm that the test suite toggles between RED and GREEN. (You will need to save the file between toggles.)

8.3 Logging out

As discussed in Section 8.1, our authentication model is to keep users logged in until they log out explicitly. In this section, we'll add this necessary logout

capability. Because the "Log out" link has already been defined (Listing 8.19), all we need is to write a valid controller action to destroy user sessions.

So far, the Sessions controller actions have followed the RESTful convention of using **new** for a login page and **create** to complete the login. We'll continue this theme by using a **destroy** action to delete sessions, i.e., to log out. Unlike the login functionality, which we use in both Listing 8.15 and Listing 8.29, we'll only be logging out in one place, so we'll put the relevant code directly in the **destroy** action. As we'll see in Section 9.3, this design (with a little refactoring) will also make the authentication machinery easier to test.

Logging out involves undoing the effects of the **log_in** method from Listing 8.14, which involves deleting the user id from the session.[15] To do this, we use the **delete** method as follows:

```
session.delete(:user_id)
```

We'll also set the current user to **nil**, although in the present case this won't matter because of an immediate redirect to the root URL.[16] As with **log_in** and associated methods, we'll put the resulting **log_out** method in the Sessions helper module, as shown in Listing 8.33.

Listing 8.33: The **log_out** method.
app/helpers/sessions_helper.rb

```
module SessionsHelper

  # Logs in the given user.
  def log_in(user)
    session[:user_id] = user.id
  end
  .
  .
```

[15] Some browsers offer a "remember where I left off" feature, which restores the session automatically, so be sure to disable any such feature before trying to log out.

[16] Setting **@current_user** to **nil** would only matter if **@current_user** were created before the **destroy** action (which it isn't) *and* if we didn't issue an immediate redirect (which we do). This is an unlikely combination of events, and with the application as presently constructed it isn't necessary, but because it's security-related I include it for completeness.

```
  .
  # Logs out the current user.
  def log_out
    session.delete(:user_id)
    @current_user = nil
  end
end
```

We can put the `log_out` method to use in the Sessions controller's `destroy` action, as shown in Listing 8.34.

Listing 8.34: Destroying a session (user logout).
app/controllers/sessions_controller.rb

```
class SessionsController < ApplicationController

  def new
  end

  def create
    user = User.find_by(email: params[:session][:email].downcase)
    if user && user.authenticate(params[:session][:password])
      log_in user
      redirect_to user
    else
      flash.now[:danger] = 'Invalid email/password combination'
      render 'new'
    end
  end

  def destroy
    log_out
    redirect_to root_url
  end
end
```

To test the logout machinery, we can add some steps to the user login test from Listing 8.24. After logging in, we use `delete` to issue a `DELETE` request to the logout path (Table 8.1) and verify that the user is logged out and redirected to the root URL. We also check that the login link reappears and that the logout and profile links disappear. The new steps appear in Listing 8.35.

Listing 8.35: A test for user logout (and an improved test for invalid login).
GREEN
test/integration/users_login_test.rb

```
require 'test_helper'

class UsersLoginTest < ActionDispatch::IntegrationTest

  def setup
    @user = users(:michael)
  end

  test "login with valid email/invalid password" do
    get login_path
    assert_template 'sessions/new'
    post login_path, params: { session: { email:    @user.email,
                                          password: "invalid" } }
    assert_not is_logged_in?
    assert_template 'sessions/new'
    assert_not flash.empty?
    get root_path
    assert flash.empty?
  end

  test "login with valid information followed by logout" do
    get login_path
    post login_path, params: { session: { email:    @user.email,
                                          password: 'password' } }
    assert is_logged_in?
    assert_redirected_to @user
    follow_redirect!
    assert_template 'users/show'
    assert_select "a[href=?]", login_path, count: 0
    assert_select "a[href=?]", logout_path
    assert_select "a[href=?]", user_path(@user)
    delete logout_path
    assert_not is_logged_in?
    assert_redirected_to root_url
    follow_redirect!
    assert_select "a[href=?]", login_path
    assert_select "a[href=?]", logout_path,      count: 0
    assert_select "a[href=?]", user_path(@user), count: 0
  end
end
```

(Now that we have **`is_logged_in?`** available in tests, we've also thrown in a bonus **`assert is_logged_in?`** immediately after posting valid information to the sessions path. We've also added a similar assertion and the solution to

the exercise from Section 8.2.4 by adding the results of Listing 8.27.)

With the session **destroy** action thus defined and tested, the initial sign-up/login/logout triumvirate is complete, and the test suite should be GREEN:

Listing 8.36: 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. Confirm in a browser that the "Log out" link causes the correct changes in the site layout. What is the correspondence between these changes and the final three steps in Listing 8.35?

2. By checking the site cookies, confirm that the session is correctly removed after logging out.

8.4 Conclusion

With the material in this chapter, our sample application has a fully functional login and authentication system. In the next chapter, we'll take our app to the next level by adding the ability to remember users for longer than a single browser session.

Before moving on, merge your changes back into the master branch:

```
$ rails test
$ git add -A
$ git commit -m "Implement basic login"
$ git checkout master
$ git merge basic-login
```