Capybara, pop up windows and the new PayPal sandbox

April 27, 2013 brafales, 0

This past weeks we have been doing a massive refactoring of our testing suite at work to set up a nice CI server setup, proper factories, etc. Our tool-belt so far is basically a well known list of Rails gems:

- Factory Girl for factories.
- RSpec as a testing framework (although we'll switch back to Test::Unit soon).
- Capybara for integration testing.

For the CI server we decided to use a third party SaaS as our dev team is small and we don't have the manpower nor the time to set it up ourselves, and we went for CircleCI, which has given us good results so far (easy to set up, in fact almost works out of the box without having to do anything, it has a good integration with GitHub, it's reasonably fast, and the guys are continuously improving it and very receptive to client's feedback).

Back to the post topic, when refactoring the integration tests, we discovered that PayPal decided recently to change the way their development sandbox works, and the tests we had in place broke because of it.

The basic workflow when having to test with PayPal involves a series of steps:

- Visit their sandbox page and log in with your testing credentials. This saves a cookie in the browser.
- Go back to your test page and do the steps needed to perform a payment using PayPal.
- Authenticate again to PayPal with your test buyers account and pay.
- Catch the PayPal response and do whatever you need to finish your test.

With the old PayPal sandbox, the login was pretty straightforward as you only needed to find the username and password fields in the login form of the sandbox page, fill them in, click the login button, and that was all. But with the new version it's not that easy. The new sandbox has no login form at the main page. It has a login button which you have to click, then a popup window is shown with the login form. In there you have to input your credentials and click on the login button. Then this popup window does some server side magic, closes itself and triggers a reload on the main page, which will finally show you as logged in.

There's probably a POST request that you can automatically do to simplify all this, but PayPal is not known as *developer documentation friendly* so I couldn't find it. As a result, we had to modify our Capybara tests to handle this new scenario. As we've never worked with pop up windows before I thought it'd be nice to share how we did it in case you need to do something similar.

The basic workflow is as follows:

- Open the main PayPal sandbox window.
- Click on the login button.
- Find the new popup window.
- Fill in the form in that new window.
- Go back to your main window.
- Continue with your usual testing.

Calendar

April 2013

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					
« Mar						

Recent Posts

- Capybara, pop up windows and the new PayPal sandbox
- ARM assembler in Raspberry Pi – Chapter 12
- ARM assembler in Raspberry Pi – Chapter 11
- ARM assembler in Raspberry Pi – Chapter 10
- ARM assembler in Raspberry Pi – Chapter 9

Recent Comments

rferrer on ARM assembler in Raspberry Pi – Chapter 11

Einstieg in Pi-Assembler | ultramachine on ARM assembler in Raspberry Pi – Chapter 1

Fernando on ARM assembler in Raspberry Pi – Chapter 7

Loren Blaney on ARM assembler in Raspberry Pi – Chapter 11

ห.ร.ม. ด้วยภาษา Assembly บน Raspberry Pi | Raspberry Pi Thailand on ARM assembler in Raspberry Pi – Chapter 9

Tags

.net activerecord ajax apple archlinux arm assembler bind branches C# dhcp firebug firefox function function call functions gadgets html indexing modes ipod Java javascript jquery linux mac os mac os x MVC networking parallels pi programming tips rails raspberry ruby ruby on rails security software sports sql server subversion tips and tricks tools ubuntu visual studio xmonad

This assumes you are using the `Selenium` driver for Capybara. Here's the code we used to get this done:

```
describe "a paypal express transaction", :js => true do
  it "should just work" do
    # Visit the PayPal sandbox url
    visit "https://developer.paypal.com/"

    # The link for the login button has no id...
    find(:xpath, "//a[contains(@class,'ppLogin_internal cleanslate scTrack:ppAccess-login

    # Here we have to use the driver to find the newly opened window using it's name
    # We also get the reference to the main window as later on we'll have to go back to i
    login_window = page.driver.find_window('PPA_identity_window')
    main_window = page.driver.find_window('')

    # We use this to execute the next instructions in the popup window
    page.within_window(login_window) do
      #Normally fill in the form and log in
      fill_in 'email', :with => "<your paypal sandbox username>"
      fill_in 'password', :with => "<your paypal sandbox password>"
      click_button 'Log In'
    end

    #More on this sleep later
    sleep(30)

    #Switch back to the main window and do the rest of the test in it
    page.within_window(main_window) do
      #Here goes the rest of your test
    end
  end
end
```

Now there is an important thing to note on the code above: the `sleep(30)` call. By now you may have read on hundreds of places that using `sleep` is not a good practice and that your tests should not rely on that. And that's true. However, PayPal does a weird thing and this is the only way I could use to make the tests pass. It turns out that after clicking the *Log In* button, the system does some behind the curtains magic, and after having done that, the popup window closes itself and then triggers a reload on the main page. This reload triggering makes things difficult. If you instruct Capybara to visit your page right after clicking the *Log In* button, you risk having that reload trigger fired in between, and then your test will fail because the next selector you use will not be found as the browser will be in the PayPal sandbox page.

There are probably better and more elegant ways to get around this. Maybe place a code to re-trigger your original visit if it detects you are still on the PayPal page, etc. Feel free to use the comments to suggest possible solutions to that particular problem.

capybara, paypal, ruby on rails, testing

ARM assembler in Raspberry Pi – Chapter 12

Leave a Reply

Your email address will not be published. Required fields are marked *

Name *

Email *

Website

Archives