Testing Your Archive

Delivering on the Promise of Persistence
@JeremyFriesen
Deck URL @ bit.ly/12VuLlp
Our Goal

An object created yesterday and accessed today should continue to be that object.

"The Ezana Stone" by A Davey @ http://flic.kr/p/5jYMhS [CC-BY]
Our Reality

The Repository Software Stack Is Ever Changing

"Construction of the Statue's pedestal" by StatueLibertyNPS @ http://flic.kr/p/ayE2sB [CC-BY]
Our Weapons

Testing

Testing

And more

Testing!

"From the Brockhaus and Efron Encyclopedic Dictionary" by Mary Margaret @ http://flic.kr/p/8biHru [CC-BY]
Our First Test

When I create it
Then it is stored as expected

"The Treachery of Images (This is Not a Pipe)" by rocor @ http://flic.kr/p/9ekGLE [CC-BY]
Our Second Test

When I didn't view it yesterday

And don't view it today

Then it is the same

Paul in Black and White, With Back Turned" by Daniel X. O'Neil @ http://flic.kr/p/dZacAJ [CC-BY]
Our Third Test

When I view it today

Then it is not changed

"Troop Inspection (Explored)" by Pascal @ http://flic.kr/p/77Xng2 [CC-BY]
Our Fourth Test

When I update it

Then it is changed as expected

"Is there any other way to prepay?" by William Clifford @ http://flic.kr/p/6m28Zv [CC-BY]
Our Test Summary

When I:
* Create it
* Leave it
* View it
* Update it

Then I test it

"test" by DaveBleasdale @ http://flic.kr/p/nNZRV [CC-BY]
Each test should be for one thing.

"one" by andrechinn @ http://flic.kr/p/5pGcyx [CC-BY]
Questions?

You want specifics?

"Questioned Proposal" by Ethan Lofton @ http://flic.kr/p/4S8uZe [CC-BY]
A Real Weapon: Capybara

Have it do your testing

http://git.io/capybara

"Capybara" by Karoly Lorentey @http://flic.kr/p/9wcG1g [CC-BY]
When I create it, then it is stored as expected.

```ruby
let(:title) { 'My Title' }

it 'is stored as expected when I create it' do
  visit new_thing_path
  within('#new_thing') do
    fill_in('Title', with: title)
    click_button('Create Thing')
  end
  pid = extract_pid_from_path(page.current_path)
  expect(class: Thing, pid: pid).to( 
    have_valid_persistence(title: title)
  )
end
```
Our Third Test - Redux

When I view it today, then it is not changed.

```ruby
let(:title) { 'My Title' }
it 'is not changed when I view it' do
  thing = Thing.create!(title: title)
  expect{
    visit thing_path(thing)
  }.to_not change_persistence(thing)
end
```
When I update it, then it is changed.

```ruby
let(:title) { 'My Title' }
let(:new_title) { title + ' New' }
it 'is changed when I update it' do
  thing = Thing.create!(title: title)
  expect {
    visit edit_things_path(thing)
    within('.edit_things') do
      fill_in('Title', with: new_title)
      click_button('Update Thing')
    end
  }.to change_persistence(thing).
  from(title: title).
  to(title: new_title)
end
```
Our First Test - Redux

When I change a translation layer
And I create the object
Then it is stored as expected.

The first test as written hopefully should be caught if there is a problem.
When I change a translation layer
And I view the object today
Then it is not changed.

This sounds like a case where you will want to clone a segment of the production persistence and index, and use that for a custom test.

After all Fedora Futures!
Our Fourth Test - Redux²

When I change a translation layer
And I update the object
Then it is changed.

Again you will want a clone of your production system to run a custom test.
But What About Second Test?

When I didn't view it … then it is the same.

Run fixity checks against your repo. Log them?

"The Fellowship of the Rings" - New Line Cinema, used without permission
Hydra Capybara Walkthrough

"A Hydra-powered Rails project built with the goal of helping you write your first set of tests. You can follow the changes to the code by replaying the commit history."

Available @ http://git.io/MkzQyA

https://github.com/jeremyf/hydra-capybara-walkthrough
Capybara without a Rails/Rack App

"Normally Capybara expects to be testing an in-process Rack application, but you can also use it to talk to a web server running anywhere on the internet"

capybara @ http://git.io/_tiIyg
capybara-mechanize @ http://git.io/C4wlwQ

https://github.com/jnicklas/capybara#calling-remote-servers
Spend Time to Automate Your Tests

Remember, as a software developer, anything you do on a computer can almost certainly be automated.

If what you are doing today will be useful tomorrow…encode it!

Bad Luck Brian meme, created @ quickmeme.com
Thank You

Jeremy Friesen
Project Application Developer
Hesburgh Library
University of Notre Dame

Slide Available @ http://bit.ly/12VuLLp

@jeremyfriesen
ndlib.github.io - a blog for helping Libraries code better