Functional Test Refactoring in Web Test Automation

Zhimin Zhan
Agile Test Coach, AgileWay
“Queensland Health's bungled new payroll system was not properly tested”

- QLD Auditor-General Report 29/06/2010

"In order to complete fix it, it will take a couple of years."
- Premier Anna Bligh 20/10/2010

"Government to spend $209 million fixing Queensland Health payroll system" - 2011-1
Why? No regression testing!

"Our wonderful CorpTech has done another 'system fix' and that has created all these other problems!!!!"  
- payroll client service officer, 2011-07-23

“The only practical way to manage regression testing is to automate it.”

- Steve McConnell, software legend

Test Automation - Reality Check

“Automated testing through the GUI is intuitive, seductive, and almost always wrong!”
- Robert C. Martin, 2009-09-30

“UI Test Automation Tools are Snake Oil”
- Michael Feathers, 2010-01-04

“For 95% of all software applications, automating the GUI is a waste of time.”
- Alan Page, is one of Microsoft's first Test Architects

http://blog.objectmentor.com/articles/2010/01/04/ui-test-automation-tools-are-snake-oil
IDT Survey 2007

"73% of survey respondents believe Automated Software Testing is beneficial but few automated!"

“Regression testing phases as most time consuming”

Why?
- Lack of Time
- Lack of Budget
- Lack of Expertise

http://journal.thedacs.com/issue/43/90
Test Automation Camel

Learning Curve: Steep

Out of Reach: Expensive

Hard to maintain

Test Refactoring

Long feedback loop

Continuous Integration

Graphic from 'http://www.freevectordownload.com', License: Creative Commons Attributes 3.0
Create your first test
< 2 minutes
Get requirement
Create new project
Record in Firefox
Run test in IE
(pass and fail)
# Comments start with '#'

story “[01] User can login” do
  open_browser “http://travel.agileway.net”
  text_field(:name,"userName").set "agileway"
  enter_text("password", "test")
  click_button("Sign In")
  page_text.should contain("Login OK!")
end

# Next Story Test
Record/Playback != Automation

“Record/Playback scripts are notoriously costly from a maintenance perspective.”
- ‘Agile Testing’ p315

“Record/playback testing tools should be clearly labeled as ‘training wheels’”
- Jason Huggins, Creator of Selenium
Assume your web site has 100 web pages, and you recorded/developed 100 automated test cases. Now programmer made two simplest changes on Page 2 and 39.

Question: To make these 100 tests pass again, how would you do it? how long will it take?
Maintainable automated test scripts
How do we talk about web sites everyday?

You shall see account summary page on there, click select pay by credit card. Enter your credit card details on payment page. Record receipt number on confirmation page.
test "[02] Book Flight" do
  enter_text("user_session[login]", "agileway")
  enter_text("user_session[password]", "test")
  click_button("Sign in")
  select_option("fromPort", "Brisbane")
  select_option("toPort", "Sydney")
  click_button("Continue")
  enter_text("passengerFirstName", "Zhimin")
  enter_text("passengerLastName", "Zhan")
  click_button("Confirm")
  page_text.should.include("Brisbane to Sydney")
end
Using Page Objects

test "[02] Book Flight" do
  home_page = expect_page Homepage
  home_page.enter_user "agileway"
  home_page.enter_password "test"

  flight_page = home_page.click_signin
  flight_page.select_from "Brisbane"
  flight_page.select_to "Sydney"

  passenger_page = flight_page.click_next
  passenger_page.enter_firstname "Zhimin"
  passenger_page.enter_lastname "Zhan"
  passenger_page.click_next

  expect_page ConfirmPage
  assert_text_present("Brisbane to Sydney")
end
Demo: Update Test Scripts with Page Objects

Programmer changed the home page
TextField ‘login’ now called ‘username’

What will happen to our automated test cases ...
Application changed (again)
Multiple test scripts failed!
Updated in seconds in TestWise
With Page-Objects
Functional Test Refactoring
Functional Test Refactoring

A step by step process to refine test scripts to make it easy to read, concise, and more importantly easy to maintain.

**DRY**, Don’t Repeat YourSelf

- Recorded Test Scripts → Refactor → Maintainable Test Scripts
Test Refactoring Benefits

- Optimising test scripts systematically
  - easy to maintain
  - easy to read
  - simple
  - fast
  - intelligent
- Team consensus
Next-Gen functional testing tool

- Agile Alliance workshop on Next-Generation Functional Testing (Oct 2007), Envision:
  - We are lacking IDE that facilitate things like:
    - refactoring test elements
    - command completion
    - incremental syntax validation
    - keyboard navigation
    - debugging

- Michael Longin’s Agile 2009 Wrap up:
  - IDE’s are the next step in automated tools
TestWise IDE

Test Folder

Test Scripts

Test execution completed.

Version 6: Refactor 7: Find 8: TCDO 9: User Story
Next-Gen Test Tool: TestWise

Testing Tool Spectrum (Paul King, assert.au)

Utilities
- All Pairs
- Combinations
- Polyglot languages
- Logic programming
- Threads
- Parallel concurrency libraries
- Data-driven libraries
- Networking libraries
- XML Processing
- Read/write files
- Excel / Word / CSV
- Reporting, Logging

Runners
- Native Groovy, JUnit, TestNG, Spock, EasyB, JBehave, Cucumber, Robot Framework, Slim

Web Drivers
- WebTest
- WebDriver
- JWebUnit
- Tellurium
- Selenium
- HtmlUnit
- Watij
- HttpBuilder
- Cyberneko

Database Drivers
- DbUnit
- DataSets
- SqUnit
- groovy.sql
- JPA
- JDO
- BigTable
- JDBC

SOAP / REST Drivers
- GroovyWS
- XML-RPC
- CXF
-Axis2
- JAX-WS
- JAX-RS

Other Drivers
- FEST
- Email
- FTP
- AntUnit
- Telnet
- SSH
- Exec

Copyright 2009: Lisa Crispin
Refactorings
1 - Extract Function

You have long logically grouped test steps performing a common operation

Extract them into a reusable function
test “case 1” do
  enter_text("userName", "david")
  enter_text("password", "four")
  click_button("Sign In")
end

def login(username, password)
  enter_text("userName", username)
  enter_text("password", password)
  click_button("Sign In")
end

test “case 1” do
  login("david", "four")
end

test “case 2” do
  enter_text("userName", "john")
  enter_text("password", "five")
  click_button("Sign In")
end

test “case 2” do
  login("john", "five")
end
Extract Function - Benefits

- DRY
- Concise
- Reusable
- Easy to maintain
- DSL, use own words
Extract Function - Demo
Refactoring
- Extract to Function
Don’t forget to run immediately after refactoring
2 - Move Function to Helper

You have use same functions defined in separate test scripts files

Move the function to the shared helper
Move to Helper - Example

# Test Script 1
```ruby
def login(username, password)
    enter_text("userName", username)
    enter_text("password", password)
    click_button("Sign In")
end

test "case 1" do
    login("david", "four")
end
```

# Test Script file 2
```ruby
# test_helper.rb
def login(username, password)
    enter_text("userName", username)
    enter_text("password", password)
    click_button("Sign In")
end

test "case 1" do
    login("john", "five")
# ...
end
```

# Test Script 1
```ruby
test "case 1" do
    login("david", "four")
    # ...
end
```

# Test Script 2
```ruby
test "case 2" do
    login("john", "five")
    # ...
end
```
Move to Helper - Benefits

- DRY
- Concise
- Auto Complete (with tools support)
Move to Helper - Demo
Refactoring
- Move to Helper
3 - Move

You have same set beginning or ending test steps in multiple test cases

Move common test steps into shared sections that run before/after each test case
test "Change password" do
  login_as("agileway")
  change_password("old", "new")
  login_as("agileway", "new")
  logout
end

test "Change address" do
  login_as("agileway")
  change_address("city", "Sydney")
  verify_address("Sydney")
  logout
end

before(:each) do
end

after(:each) do
end

test "Change address" do
  change_password("pass", "new")
  login_as("agileway", "new")
end

test "Change address" do
  change_address("city", "Sydney")
  verify_address("Sydney")
end
Move - Benefits

- DRY
- Concise
- Focused Test
Move - Demo
Refactoring
- Move
The name of a function does not reveal its purpose

Change function name
test script file

test "case 1" do
  enter_text("userName", "david")
  enter_text("password", "test")
  click_button("Sign In")
end

# In page class: HomePage

def enter_user(username)
  enter_text("userName", username)
end

def enter_password(pass)
  enter_text("password", pass)
end

def click_signin
  click_button("Sign In")
end
Extract Page Class - Benefits

- DRY
- Concise
- Reusable
- Easy to maintain
- DSL, easy to read
- Auto complete via tool support
Page-Objects
- Extract Page Refactoring
5 - Introduce Page Object

Want to use a predefined Page class

Introduce a page object
Introduce Page Object: Example

```
expect_page HomePage

home_page = expect_page HomePage

home_page.operation
```
Auto Complete
- helper function
- page function
Navigation
6 - Rename

The name of a function does not reveal its purpose

Change function name
# In Page Class: HomePage

def click_signin
    click_button("Sign In")
end

# test script file

test “case 1” do
    home_page = expect_page Homepage
    home_page.enter_user “david”
    home_page.enter_password “test”
    home_page.click_signin
end

# another test script file

test “case 2” do
    home_page = expect_page Homepage
    home_page.enter_user “david”
    home_page.enter_password “test”
    home_page.click_signin
end

# In Page Class: HomePage

def click_login
    click_button("Sign In")
end

# test script file

test “case 1” do
    home_page = expect_page Homepage
    home_page.enter_user “david”
    home_page.enter_password “test”
    home_page.click_login
end

# another test script file

test “case 2” do
    home_page = expect_page Homepage
    home_page.enter_user “david”
    home_page.enter_password “test”
    home_page.click_login
end

All references of this function updated
Rename — Benefits

- Keep test scripts readability
  - Quite Important ★★★
Rename - Demo

Personal Details ( * mandatory fields)

Title:  First name: *  Surname *

Title:  First name: *  Last name *
Rename

surname => last_name
Test Automation in a Real Project

2 and half years

5—7 Users

250 Test Cases, 94 Page Class

Running on two machines concurrently

2 Hours

200,000+ test case executions
Thanks

Questions?

Happy Functional Testing
http://testwisely.com