

Practical Applications of Agile Web Testing Tools

Titus Brown and Grig Gheorghiu
PyCon 2008, Chicago

Web UI Testing with Selenium

- functional/acceptance testing at the UI level for Web applications
 - uses an actual browser driven via JavaScript
 - AJAX testing
 - browser compatibility testing
 - several components
 - Selenium Core
 - Selenium RC
 - Selenium IDE
 - Selenium Grid
-
-

Selenium Core

- needs to be deployed on same server with Web app under test (due to JS security restrictions)
 - tests and test suites written HTML tables
 - tests consist in actions (open, click, type), verifications and assertions
 - AJAX-specific actions are provided (waitFor commands)
 - can be slow and hard to automate
 - test code looks ugly, hard to reuse
-
-

Selenium Remote Control (RC)

- uses Java-based HTTP proxy server
 - browser polls the server for commands to execute
 - bindings available to all major scripting languages
 - allows you to write tests in a real programming language
 - allows tests to run as part of a Continuous Integration process
 - Google runs 51K SelRC tests daily, 4% of which fail
-
-

Selenium IDE

- record/playback tool available as Firefox plugin
 - greatly simplifies the creation of tests
 - can export HTML-based tests into Selenium RC-type tests (Python, Ruby, etc.)
 - version 1.0 will capture AJAX-specific actions (drag/drop etc.)
 - invaluable if you are using Selenium heavily
 - other useful Firefox plugins: XPath Checker, XPather, Firebug
-
-

Selenium – Lessons Learned

- GUI tests are brittle
 - use HTML ids for elements if you can
 - XPath expressions can be very complicated (UI Map project tries to alleviate that)
 - use a tool such as twill for non-JavaScript testing, use Selenium for heavy JS/AJAX testing
 - WebDriver will be integrated into Selenium (browser-specific plugins)
 - Selenium meetup @ Google:
<http://youtube.com/watch?v=EDb8yOM3Vpw>
-
-

Continuous Integration with Buildbot

- continuous integration
 - automate time-consuming tasks
 - important for the immediate public feedback it gives you about changes you've made
 - the more often you build and test your software, the quicker you will discover and fix bugs
 - buildbot is based on Twisted
 - pretty hard to configure, but works really well once you have it up and running
 - can be deployed behind Apache for access control purposes
-
-

Continuous Integration with Buildbot (cont.)

- master process kicks off build-and-test process on configured slaves (via scheduler, SVN notifications)
 - master.cfg written in Python, easy to extend
 - you get the most value out of the CI process if you have build slaves running on as many of the OSes/platforms that you plan to support for your app
-
-

Continuous Integration with Buildbot (cont.)

- run as many types of testing as possible
 - unit tests (nose)
 - acceptance tests (FitNesse)
 - functional tests (twill)
 - UI tests (Sel RC)
 - coverage and profiling
 - deployment tests (egg creation/installation)
 - the more aspects of your application you cover, the better you are protected against unexpected breakages
-
-

Continuous Integration with Buildbot – Lessons Learned

- running tests as specific user under a different environment will uncover all kinds of issues
 - OS versions
 - Python versions
 - package versions
 - hard-coded paths
 - log locations
 - environment variables
-
-