Agile Web Application Testing

Technologies and Solutions

V. Narayan Raman
Tyto Software

Goals

Rapid feedback on the quality of software

Problem in Web App Testing

- Many Browsers
- Many Operating Systems
- Browsers have quirks
- Browser technologies are evolving at an amazing pace

Functional or Unit Testing?

- Functional tests
  - How the end user engages with the system
  - Very important
- Unit tests
  - Does not matter to the end user
  - Necessary from a dev perspective

Functional ~~or~~ AND Unit Testing

Automation vs. Manual

- Automation
  - Can rapidly go over known scenarios and paths
- Manual
  - Can be creative and go over new scenarios and explore new ways of interacting with the system

Automation is a time saving tool for Manual testers

Agile

- Ability to rapidly adapt to changes in business requirement
  - Automation scripts can break due to evolution of web application
  - Good engineering practices are required to ensure effectiveness

Engineering practices

- Frequent check-ins
- Frequent releases
- A good continuous integration set-up
  - CruiseControl, Jenkins etc.
- Ensure Automation scripts are green before delivering the build to QA for testing
  - Needs good interaction between Testers and Devs

Automation Tools

- Everything on the desktop
  - Good for broad level testing
  - Ineffective for complex web applications

- Only web applications
  - Good for web app testing
  - Tools: Sahi, Selenium, Watir

Technology choices of Tools - Factors

- Identification of elements
- Event Simulation
- Reliability of playback
- Library Dependencies
- Recorders/Object Spys

Identification of elements

- Ids, names and other DOM properties
  - Best if you can have them. Not always possible though
- XPaths
  - A very popular but amazingly bad way of writing tests
- CSS Selectors
  - Better than XPaths but still depends too much on code structure
- User Interface Relations
  - Best way to identify elements without depending on code structure

Examples

- DOM
  - _textbox(“username”)
- XPath
  - "/html/body/table[@id='rootfragment']/tbody/tr[2]/td/table/tbody/tr[2]/td[4]/table/tbody/tr[1]/td/table/tbody/tr[1]/td/div/table/tbody/tr/td[2]/table/tbody/tr/td[1]/strong“

Examples

- CSS Locator
  - "css=div#myID > a:contains(Confirm Transaction)“

- User Interface Relations

```
_button(“Delete”, _near(_cell(“Ram”)))
_cell(0, _near(_div("My product one")), _under(_div("Cost")))
```

Good Identifiers

- Should be resilient to structural changes in UI
  - XPaths and CSS Selectors are dependent on code structure
- Should remain constant even when moved across frames/iframes
- Should be able to resolve one element with respect to another
  - _checkbox(0, _under(_tableHeader(“Choose”)))

Event Simulation

- OS level events
  - Theoretically more correct
  - Needs focus of window/element (prevents reliable parallel playback)
- Browser level events
  - Vary between browsers
  - Does not need focus of browser

Reliable Playback

- Problem
  - Variable load times for AJAX, Page/Frame/Iframe
- Solutions
  - Explicit waits
    - _wait(30000)
    - Bad practice
  - Explicit conditional waits
    - _wait(30000, _isVisible(element));
    - Needs more code and knowledge of application
  - Implicit Waits
    - Tool decides how much to wait
    - Much better and more reliable

Library Dependencies

Usage of common development libraries for testing
Eg. JQuery, Sizzle

- Problem
  - Conflict between tool's version and web application's version
  - 2 web apps with different library versions cannot be tested with the same tool
- Solution
  - Avoid dependence on libraries which are used in your web app development

Recorder/Object Spy

- Recorder
  - Nice to have
  - But scripts should be refactored
- Object Spy
  - Absolute must have
  - For rapid feedback and trouble shooting across browsers

Automation Approach 1
Protocol Level

- Protocol level request response validation
  - May use a proxy
  - Cannot simulate javascript events reliably
  - Does not work for modern AJAX applications with lots of snippets of request responses

Automation Approach 2
Proxy Based Javascript Injection

- Javascript injection via Proxy
  - Uses technologies like Proxy and Javascript which are guaranteed to be implemented by all browsers (with minor differences)
  - Works across browsers and can easily support a new browser
  - Con: Needs configuration of Proxy, SSL certificate management

Automation Approach 3
Browser Extensions

- Very tedious to write extensions for every browser on every OS
- Maintenance problem for project developers
- Common libraries for javascript needs to be injected similar to proxy based approach

Comparison of common Tools

Criteria	Sahi	Selenium 1	Selenium 2
Identification	DOM, UI Relations (XPath, CSS Selector also possible)	DOM, XPath, CSS Selector	DOM, XPath, CSS Selector
Event Simulation (Predominant)	Browser	Browser (but does not simulate all events)	Native
Waits (Predominant)	Implicit	Explicit + Conditional	Half way
Library dependencies	None by default. Needed for XPath and CSS Selector	For XPath and CSS Selector	For XPath and CSS Selector
Recorder/Object Spy	Yes on all browsers	Yes on Firefox only	In development for Firefox only
Automation approach	Proxy based	Proxy based + Plugin based	Native extensions for browsers

Other features to look out for

- Integration with build tools and CI systems
- Parallel playback
  - Saves time
- IDE support (Real ones, not Se IDE)
- Refactorable test scripts
- Easily understandable and learnable by testing team
- Inbuilt Reporting

Note to Developers

- A testing tool is used mostly by testers
  - Remember how long you took to be good at any programming language.
  - Don’t force your programming language onto your testers. It will hamper productivity
- Really, don’t think like a developer when testing
  - Bad practices like XPath usage have come in because of thinking in code
  - Think like the End User

Things to check out

- Tools
  - Sahi, Selenium, Watir
- Behavior Driven Development (BDD)
  - Cucumber, Behat, EasyB, Twist
- Performance and Load Testing
  - PushToTest TestMaker
- Continuous Integration
  - Cruise Control, Jenkins (Hudson)

Q & A

Thank You

Twitter: @narayanraman

Website: http://sahi.co.in/