УДК 004.4'22

АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ WEB-СТОРІНОК ЗА ДОПОМОГОЮ ДРАЙВЕРА БРАУЗЕРА SELENIUM WEBDRIVER

Богач І.В., Кравець А.Ю.
Вінницький національний технічний університет

В дані роботі наведені основні поняття автоматизованого тестування. Виділені загальні етапи тестування, та основні етапи швидкого тестування web-сайту. Були визначені основні переваги роботи з Selenium WebDriver, охарактеризовані його межі роботи. Визначені перспективи використання даного драйвера браузера, можливі програмні стеки для його найефективнішого впровадження.

Ключові слова: тестування, автоматизоване тестування, тестовий сценарій, драйвер браузера, Selenium WebDriver, web-сайт.

Постановка проблеми. Створення будь-якої сторінки, навіть досвідченими розробниками, неможливе без помилок різного характеру, які можуть погіршити роботу сайту. За витратами часу і людських ресурсів найзатратнішими є етапи розробки, пов'язані з пошуком помилок у готових продуктах. Незважаючи на те, що зусилля, необхідні для внесення невеликих змін, як правило, мінімальні, вони можуть вимагати чималих зусиль для перевірки якості зміненої програми. І при досягненні певного критичного об'єму робіт по тестуванню web-сторінки за короткий проміжок часу одним з очевидних рішень стає автоматизація процесу тестування, головною проблемою якого є відсутність універсального засобу автоматизації тестування.

Аналіз останніх досліджень і публікацій. З огляду інформаційних джерел видно, що на ринку присутня досить велика кількість систем автоматизованого тестування загалом та web-ресурсів зокрема [1]. Також можна побачити результати досліджень щодо використання різних інструментів автоматизації тестування в реальних проектах [2]. В роботі [3] було виділено основні патерни автоматизованого функціонального тестування користувацьких інфтерфейсів, які можна обрати за основу для загального функціонального тестування. А також у роботі [4] було визначено модель процесу автоматизованого тестування в умовах багато продуктових компаній.

Мета роботи. Метою роботи є зменшення часу, що витрачається на проведення стандартних частовживаних тестів при розробці web-сторінок.

Виклад основного матеріалу дослідження. Тестування – техніка контролю якості, що перевіряє відповідність між реальною і очікуваною поведінкою сторінки завдяки кінцевому набору тестів, які обираються певним чином [5].

Якість не є абсолютною, це суб'єктивне поняття. Тому тестування, як процес своєчасного виявлення помилок та дефектів, не може повністю забезпечити коректність роботи сторінки. Воно тільки порівнює стан і поведінку продукту зі специфікацією. Зазвичай, поняття якості обмежується такими поняттями як коректність, надійність, практичність, безпечність, але може містити більше технічних вимог, котрі описані у стандарті ISO 9126. Склад та зміст супутньої документації процесу тестування визначається стандартом IEEE 829-1998 Standard for Software Test Documentation.

Тестування вважається успішним, якщо знайдено дефект або помилку, і вони відразу усуваються. Ступінь тестованості визначається критерієм покриття системи тестами, перевірки всіх можливих шляхів виконання програм й імовірності припущення стосовно того, що може з'явитися збій або помилкова ситуація в системі.

Загалом будь-яке тестування продукту, зазвичай, складається з трьох основних етапів:

1. Вивчення та аналіз предмета тестування;
2. Планування процесу тестування (складання планів, тестів, наборів даних) готового продукту;
3. Виконання тестування:

– проведення тестування компонентів повторного використання і патернів як основних об'єктів продукту;

– генерація необхідних тестових сценаріїв, що відповідають середовищу виконання продукту;

– верифікація правильності реалізації системи і валідація реалізації вимог до готового продукту;

– збирання даних про відмови, помилки і виявлені непередбачені ситуації при виконанні продукту;

– підготовка звітів за результатами тестування й оцінка характеристик продукту.

Зв'язавши цикл тестування з циклом розробки, ми бачимо, що вивчення та аналіз предмета тестування починаються перед затвердженням специфікації (на завершення стадії «Розробка дизайну продукту і створення документації») і продовжується на стадії «Кодування»; планування тестування відбувається на стадії «Кодування»; виконання тестування відбувається на стадії «Виконання тестування і ремонт багів» [6-7].

Важливо відмітити, що показаний зв'язок між циклом розробки і циклом тестування – це всього лише типова модель взаємодії процесів, яка не обов'язково виконується на практиці.

Зазвичай в реальних умовах швидке тестування проводиться за такими етапами:

1. Підготовчі роботи по вивченню документації.
2. Функціональне тестування. Найбільш тривалий етап перевірки ресурсу, автоматизація якого є головною метою. Суть цього процесу полягає у перевірці всього описаного функціоналу:

– роботи всіх обов'язкових функцій сайту;

– працездатності форм користувача на сайті (введення тексту, чисел, використання маски, робота з незаповненими полями, довжина символів, що вводять, коректний робота чекбоксів, комбобоксов, radio buttons, логічність установок «за замовчуванням», зворотній зв'язок, додавання коментаря в блог);

– роботи пошуку (включаючи релевантність результатів);

– гіперпосилань, пошук неробочих посилань;

– підвантаження файлів на сервер;

– працездатності лічильників, встановлених на сторінках сайту;

– перегляд на відповідність вмісту сторінок сайту вихідного контенту, наданого замовником.

– перевірка баз даних (пошук, додавання інформації, редагування, видалення, перевірка на дублювання інформації).

– наявності секретних частин (робота з паролями, передача даних, захист)

3. Тестування верстки – при перевірці верстки першим ділом тестувальник перевіряє розташування елементів, відповідність їх позицій наданим макетів, а так само перевіряє оптимізацію зображень і графіки. Сюди входить перевірка на єдність дизайну. Тут же має сенс оцінювати сумісність з дизайном звуків, малюнків та анімації, а також перевірити чи має місце єдність відображення при використанні інших екранних розширень і глибин кольору.

4. Перевірка валідності коду та кросбраузерності.

5. Тестування на дружність проводиться для оцінки зручності продукту у використанні, заснованої на залученні користувачів в якості тестувальників і аналізі отриманих результатів.

6. Тестування безпеки. На даній стадії тестування фахівець перевіряє чи немає у користувачів доступу до службових, закритих сторінок а так само проводить перевірку захисту всіх критично важливих сторінок (наприклад, розділу адміністрування сайту) від зовнішнього впливу.

7. Тестування продуктивності сайту – проводиться з метою визначення швидкодії сайту або його частини під певним навантаженням. Тестування продуктивності включає в себе такі види тестування: тестування навантаження, швидкодії [8-10].

Автоматизоване тестування забезпечує переваги, які можуть підвищити ефективність роботи відділу тестування в довгостроковій перспективі. За допомогою автоматизованого тестування можна:

– проводити частіше регресійне тестування;

– швидко надавати розробникам звіт про стан продукту;

– отримати потенційно нескінченне число прогонів тестів;

– забезпечити підтримку Agile і екстремальних методів розробки;

– зберігати сувору документацію тестів;

– виявити помилки, які були пропущені на стадії ручного тестування.

Найчастіше інструменти автоматизованого тестування відповідають лише декільком переліченим пунктам. Задля підвищення ефективності тестування та збільшення його використання необхідний універсальний засіб, який базувався б на основних патернах для тестування та мав вигляд загального фреймворку. Так як вже було зазначено раніше, функціональне тестування є найтривалішим етапом тестування web-ресурсу то основним напрямком роботи фреймворку є саме функціональне приймальне тестування.

Оптимальним шляхом вирішення цієї задачі було обрано бібліотеку драйверів браузера Selenium WebDriver, на основі якої буде розроблений фреймворк, оскільки вона має ряд переваг над усіма іншими аналогічними вже вбудованими бібліотеками в інструменти тестування. Це підтверджується її використанням у великих компаніях як Google для тестування своїх сервісів і продуктів.

Selenium WebDriver (Selenium 2.0) – це програмна бібліотека для управління браузерами. Часто вживається також більш коротка назва WebDriver.

Іноді кажуть, що це «драйвер браузера», але насправді це ціле сімейство драйверів для різних браузерів, а також набір клієнтських бібліотек на різних мовах програмування (рисунок 1), що дозволяють працювати з цими драйверами.

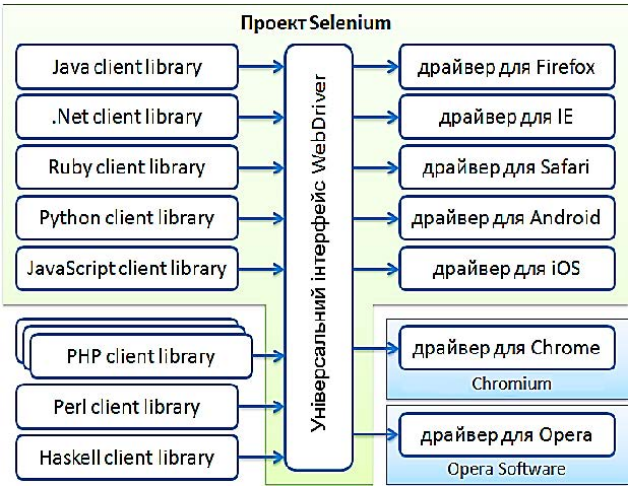

Рис. 1. Складові проекту Selenium WebDriver

WebDriver підтримує наступні браузери та операційні системи:

– Google Chrome 12.0.712.0+;

– Internet Explorer 6, 7, 8, 9, 32-бітові та 64-бітові версії;

– Firefox 3.0, 3.5, 3.6, 4.0, 5.0, 6, 7, 8, 9;

– Opera 11.5+;

– HtmlUnit 2.9;

– Android-2.3+ для телефонів і планшетів (пристроїв і емуляторів);

– iOS 3+ для телефонів (пристроїв і емуляторів) і 3.2+ для планшетів (пристроїв і емуляторів).

Аналогічна ситуація і з клієнтськими бібліотеками – в рамках проекту Selenium розробляються бібліотеки для мов Java,.Net (C #), Python, Ruby, JavaScript [11-12].

За своєю сутністю Selenium WebDriver являє собою:

– специфікацію програмного інтерфейсу для управління браузером;

– референсні реалізації цього інтерфейсу для декількох браузерів;

– набір клієнтських бібліотек для цього інтерфейсу на декількох мовах програмування.

Так як Selenium WebDriver драйвер браузера, то це програмна бібліотека, яка не має користувальницького інтерфейсу та дозволяє різним іншим програмам взаємодіяти з браузером, керувати його поведінкою, отримувати від браузера певні дані та змушувати його виконувати певні команди. Виходячи з цього визначення, видно, що WebDriver лише надає автотестам доступ до браузера. На цьому його функції закінчуються. Структурування, угруповання і запуск тестів, а також генерацію звітів про тестування, забезпечує фреймворк тестування, такий як JUnit або TestNG для Java, NUnit або Gallio для.Net, RSpec або Cucumber для Ruby. Розробка тестів ведеться в середовищі Eclipse, Intellij IDEA, Visual Studio, RubyMine. Збірка здійснюється за допомогою Maven, Gradle, Ant, NAnt, Rake. Запуск тестів за розкладом і публікацію звітів виконує сервер безперервної інтеграції – Jenkins, CruiseControl, Bamboo, TeamCity. Що обумовлює універсальність використання WebDriver для розробників та користувачів фреймворком.

Наприклад, стек для технології Java може бути таким: Jenkins + Maven + Thucydices + JUnit + WebDriver. До цього додаються ще всі можливості мови програмування Java, плюс маса плагінів для Maven і Jenkins. Також для збільшення універсаль-

ності можна запускати тести в хмарах, використовуючи який-небудь сервіс типу SauceLabs.

Найголовніша відмінність WebDriver від всіх інших драйверів полягає в тому, що це «стандартний». Оскільки організація W3C прийняла WebDriver за основу при розробці стандарту інтерфейсу для управління браузером. Зараз він перебуває в стані публічного розгляду [13].

Можливості роботи WebDriver:

– пошук елементів в основному дереві сторінки, отримання їх атрибутів та стилів;

– взаємодія з елементами: фокусування, клік, натиснення на клавіші;

– використання різних типів очікування;

– виконування дій курсором миші (при цьому реальний курсор не використовується);

– перехід за посиланнями, перемикання між вікнами браузера, робота з cookies;

– можливість збереження скріншотів;

– завантажування та скачування файлів;

– виконання скриптів на сторінці (JavascriptExecutor);

– перемикання між вкладеними фреймами.

За допомогою WebDriver не реалізується:

– власне тестування;

– складання звітів;

– робота з рівнями подій ОС, а також з діалоговими вікнами браузера (відправка на друк);

– клацання по невидимому елементу;

– перехвачування запитів [12].

Також за допомогою Selenium можна реалізовувати неперервну інтеграцію та Ajax.

Основним завданням неперервної інтеграції є автоматизація процесів розробки і тестування, з тим, щоб вони могли автоматично запускатися один або кілька разів на день замість, наприклад, одного разу на місяць ручним способом. Основна перевага використання неперервної інтеграції полягає в тому, що зміна коду інтегрується автоматично і на постійній основі. Якщо в системі відбувся збій і її створення пройшло невдало, інструментальні засоби неперервної інтеграції можуть автоматично повідомити про це.

Ajax підтримує Asynchronous JavaScript і XML; це новий термін для відносно старої технології. Основна ідея Ajax полягає в тому, що web-додаток набагато швидше реагує на дії користувача, оскільки замість всієї сторінки оновити потрібно тільки її частину.

Ajax представляє більше складнощів для web-додатків, що також відображається в тестуванні. Це відбувається тому, що Ajax використовує JavaScript і асинхронні запити HTTP для того, щоб оновити зміст сторінки. При реалізації кожен браузер має невеликі відмінності. Оскільки Selenium запускається в найпопулярніших браузерах, він створює відмінні інструментальні засоби для тестування і визначення цих відмінностей [14].

Висновки і перспективи. Отже, швидке тестування web-сайту складається з таких основних етапів: функціональне тестування, тестування верстки, безпеки сайту, валідності та продуктивності коду, тестування на дружність, на сумісність з різними браузерами і ОС. Найтривалишім та найважливішим етапом є функціональне тестування, яке можна автоматизувати за допомогою драйвера браузера Selenium WebDriver. Його переваги заключаються в тому, що він надає можливість роботи з великим набором браузерів та мов програмування та є інтуїтивно зрозумілим за рахунок мінімалізації набору команд, а також підтримує неперервну інтеграцію, технологію Ajax та відтворює усі можливі дії користувача.

Список літератури:

1. List of GUI testing tools Testing [Електронний ресурс] – Режим доступу до файлу: http://en.wikipedia.org/wiki/List_of_GUI_testing_tools
2. Результаты опроса по автоматизированному тестированию [Електронний ресурс] – Режим доступу до файлу: http://habrahabr.ru/post/255127/
3. Ремінний О. А. Патерни автоматизованого функціонального тестування користувацьких інтерфейсів / О. А. Ремінний // Інформаційні технології та комп'ютерна інженерія. – 2013. – № 3. – С. 10-15. – Режим доступу: http://nbuv.gov.ua/j-pdf/Itki_2013_3_4.pdf
4. Ремінний О. А. Модель процесу автоматизованого тестування користувацьких інтерфейсів в умовах багатопродуктових компаній / О. А. Ремінний // Вимірювальна та обчислювальна техніка в технологічних процесах. – 2013. – № 4. – С. 106-113. – Режим доступу: http://nbuv.gov.ua/j-pdf/vott_2013_4_18.pdf
5. Элфрид Дастин, Джефф Рэшка, Джон Пол. Автоматизированное тестирование програмного обеспечения – М.: Лори, 2003. – 590 с.
6. Автоматизированое тестирование [Електронний ресурс] – Режим доступу: http://habrahabr.ru/post/111292/
7. Котляров В. П., Коликова Т. В. Основы тестирования программного обеспечения. – Интернет-Университет информационных технологий, Бином. Лаборатория знаний, 2006 – 351 с. ISBN: 5-9556-0027-2, 5-94774-406-4
8. Правильное тестирование [Електронний ресурс] – Режим доступу: http://s-test.narod.ru/Articles/webtest.htm
9. Савин Р. С13 Тестирование Дот Ком, или Пособие по жестокому обра- щению с багами в интернет-стартапах. – М.: Дело, 2007 – 504 с.
10. Винниченко И. В. Автоматизация процессов тестирования. – СПб.Ж Питер, 2005 – 203 с. ISBN 5-469-00798-7
11. Selenium 2.0 и Webdriver [Електронний ресурс] – Режим доступу: http://selenium2.ru/docs/webdriver.html
12. Что такое Selenium? [Електронний ресурс] – Режим доступу: http://habrahabr.ru/post/152653/
13. Что такое Selenium WebDriver? [Електронний ресурс] – Режим доступу: http://habrahabr.ru/post/152971/
14. Автоматические приемочные тесты с Selenium? [Електронний ресурс] – Режим доступу: http://www.ibm.com/developerworks/ru/library/wa-selenium-ajax/

Богач И.В., Кравець А.Ю.
Винницкий национальный технический университет

АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ WEB-СТРАНИЦ С ПОМОЩЬЮ ДРАЙВЕРА БРАУЗЕРА SELENIUM WEBDRIVER

Аннотация
В данной работе приведены основные понятия автоматизированного тестирования. Выделенные общие этапы тестирования, и основные этапы быстрого тестирования web-сайта. Были определены основные преимущества работы с Selenium WebDriver, охарактеризованы его пределы работы. Определены перспективы использования данного драйвера браузера, возможны программные стеки для его самого эффективного внедрения.
Ключевые слова: тестирование, автоматизированное тестирование, тестовый сценарий, драйвер браузера, Selenium WebDriver, web-сайт.

Bohach I.V., Kravets A.Yu.
Vinnytsia National Technical University

AUTOMATED TESTING OF WEB-PAGES USING DRIVER OF SELENIUM WEBDRIVER

Summary
This paper presents the basic concepts of automated testing. Common stages of testing and the main stages of rapid web-site testing are specified. The main advantages of working with Selenium WebDriver are identified and its work is characterized. The prospects of the use of the browser software and possible stacks for its most efficient implementation are identified.
Keywords: testing, automated testing, test script, driver of browser, Selenium WebDriver, web-site.

УДК 004.942

МОДЕЛЮВАННЯ ОПТИЧНИХ СТРУКТУР МЕТОДОМ КІНЦЕВИХ РІЗНИЦЬ У ЧАСОВОМУ ПРОСТОРІ

Довгалець С.М., Малишевський О.М.
Вінницький національний технічний університет

В даній роботі розглянуто один з найпопулярніших методів чисельного вирішення електродинамічних задач FDTD, що вирішує електричні і магнітні поля в тимчасових і просторових доменах та дозволяє будь-яку геометрію моделі і не накладає ніяких обмежень на властивості матеріалу. Розглянуто теоретичні відомості рівнянь електромагнітних полів, визначення значень при поширені хвиль та проблеми отримання значень при граничних умовах. Запропонований метод моделювання криволінійних поверхонь розділу середовищ дозволяє моделювати поверхневі електромагнітні та плазмонні коливання без додаткових утрат точності.
Ключові слова: метод кінцевих різниць у часовому просторі, комірка Yee, електромагнітне поле, граничні умови, моделювання довільної поверхності.

Постановка проблеми. В задачах електродинаміки актуальним питанням є моделювання. Найбільш ефективним методом розрахунку математичних моделей є метод кінцевих різниць, але він вимагає значну кількість системних ресурсів. Тому для розрахунку рівнянь Максвелла у диференціальній формі використовується метод кінцевих різниць у часовому просторі (англ. Finite Difference Time Domain, FDTD), що дозволяє значно зменшити використання ресурсів комп'ютера. Він може бути з успіхом застосований для моделювання наддовгих електромагнітних хвиль в геофізиці і мікрохвиль, вирішення завдань в оптичному діапазоні (фотонні кристали, наноплазмоніка, солітони і біофотоніки). Особливо ефективним є застосування методу FDTD в тих завданнях, в яких пасують традиційні підходи, зокрема – де важлива можливість аналізу нестаціонарних процесів.

Аналіз останніх досліджень і публікацій. У виданні [1] викладені основні положення методу кінцевих різниць у часовому просторі та перспективні напрямки його використання. У роботі [2] було розроблено техніку, що реалізує явну різницеву схему другого порядку для вирішення вихрових рівнянь Максвелла у просторі та часі. Також у виданні [3] було розглянуто межі використання двохвимірного моделювання за допомогою FDTD.

Мета роботи. Метою роботи є розробка методики для отримання числових рішень проходження процесів поширення хвиль розв'язанням диференціальних рівнянь методом FDTD.

Виклад основного матеріалу дослідження. Створення пристроїв, які використовують поверхневі плазмон – поляритонні хвилі на поверхні металу для переносу інформації, вимірювання та локальної взаємодії із речовиною потребує методів чисельної електродинаміки, які враховують метал